FACULTAT D'INFORMÀTICA DE BARCELONA (FIB)
UNIVERSITAT POLITÈCNICA DE CATALUNYA (UPC) – BarcelonaTech

Reenginyeria de la plataforma RISCOSS

Grau en Enginyeria Informàtica
Enginyeria de Software

Autora: Maria Vives Montero
Director: Xavier Franch Gutiérrez
Enginyeria de Serveis i Sistemes d'Informació
Codirectora: Lidia López Cuesta
Enginyeria de Serveis i Sistemes d'Informació

24 d'Abril 2018

Resum

L’objectiu del projecte és fer l’anàlisi de l'arquitectura i tecnologies actuals, identificació d'aspectes a millorar i implementació, prova, integració i documentació de la millora de la plataforma RISCOSS (Managing Risk and Cost in Open Source Software Adoption).

Resumen

El objetivo del proyecto es realizar el análisis de la arquitectura y tecnologías actuales, identificación de los aspectos a mejorar y implementación, prueba, integración y documentación de la mejora de la plataforma RISCOSS (Managing Risk and Cost in Open Source Software Adoption).

Abstract

The objective of this project is to analyse the current architecture and technologies, identification of the aspects to be improved and implementation, testing, integration and documentation of the improvement of the RISCOSS (Managing Risk and Cost in Open Source Software Adoption) platform.

Agraïments

A en Xavier Franch Gutiérrez per la supervisió del projecte.

A la Lidia López Cuesta per la seva dedicació, supervisió, paciència i temps que ha dedicat a les reunions.

Als meus companys de feina pel seu suport i ànims a l'última fase de la carrera.

I finalment, a tota la meva família pel seu suport i per la seva confiança durant tota la carrera.

Índex

Índex taules

Índex figures

1. Introducció i contextualització

1.1 Context

Aquest projecte és un Treball Final de Grau de l'especialitat d'Enginyeria del Software de la Facultat d'Informàtica de Barcelona (Universitat Politècnica de Catalunya). És un projecte de modalitat A, que té com a objectiu fer reenginyeria de la plataforma RISCOSS (Managing Risk and Cost in Open Source Software Adoption). Concretament, el que es vol fer és anàlisis de l'arquitectura i tecnologies actuals de la plataforma RISCOSS, identificació d'aspectes a millorar (potencialment la base de dades) i implementació, prova i documentació de la millora. Aquesta plataforma va ser el resultat d'un projecte de recerca, del mateix nom, finançat per la Comissió Europea en la convocatòria FP7, durant el període 2012-2015.

RISCOSS és una plataforma web amb l'objectiu d'avaluar els riscos i costos que es deriven de l'adopció de components open source en projectes software. Aquests riscos estan relacionats amb aspectes com ara: conflictes de llicència dels components, riscos de seguretat, adopció de components amb comunitats poc estables, etc. El projecte de recerca RISCOSS va generar una plataforma basada en la recol·lecció sistemàtica de dades a partir de repositoris típics open source (Jira, GitHub, ...) i de l'anàlisi del codi mateix, que es combinen (en base a certs models lògics) per calcular el valor d'indicadors estratègics de risc, presentats als decision-makers que poden aleshores saber les conseqüències d'adoptar els components avaluats.

En el desenvolupament del projecte RISCOSS han participat Universitat Politècnica de Catalunya, Fondazione Bruno Kessler i Univ. Maastricht com a partners acadèmics i com a partners industrials Ericsson, OW2, XWiki, KPA i CENATIC.

1.1.1 Actors implicats

En aquest apartat es defineixen els actors implicats en aquest projecte, és a dir, a qui va dirigit, qui ho usarà i qui es beneficiarà dels resultats.

1.1.1.1 Director del projecte i codirectora del projecte

El director i la codirectora del projecte són en Xavier Franch Gutiérrez i la Lidia López Cuesta, respectivament. Són les persones que s'encarregaran de guiar a la desenvolupadora i d'ajudar a fer que totes les fases del projecte es facin de manera correcta i en el termini establert.

1.1.1.2 Desenvolupadora

La desenvolupadora del projecte és la Maria Vives Montero. Aquesta persona és la que s'encarregarà de fer la part tècnica (analista, desenvolupadora i tester), documentació i presentació. Totes les decisions es prendran junt amb el director i/o codirectora del projecte.

1.1.1.3 Usuaris

Aquest grup el forma qualsevol organització privada o pública, usuaris que vulguin fer ús de la plataforma i també usuaris que vulguin utilitzar el codi. Hi ha tot aquest ventall de possibilitats perquè és un software lliure i qualsevol persona pot tenir accés gratuït.

1.1.1.4 Beneficiaris

En aquest cas, els beneficiaris d'aquest projecte són:

- Els partners que han desenvolupat la plataforma perquè aquest projecte el que farà és contribuir a millorar-lo en tots els aspectes, des de la documentació fins a alguns components bàsics com ara la base de dades.
- Els usuaris (empreses, institucions públiques, etc.) perquè indirectament utilitzaran un software que tindrà actualitzacions amb millores incorporades.

1.2 Formulació del problema

En els desenvolupaments de projectes de software es prenen diferents decisions en l'àmbit d'arquitectura que més endavant potser no són la millor opció. En aquest cas, un dels partners que va participar en el desenvolupament de la plataforma (UPC) ha identificat dos possibles punts de millora, la base de dades i les llicències d'algun dels components externs que es fan servir. L'abast del TFG és realitzar la millora a la BD (base de dades).

1.2.1 Base de dades

Actualment, no hi ha cap tipus de document on s'indiqui el disseny que es va fer, com s'ha d'accedir a les dades, etc. Tampoc s'ha rebut resposta per part de la persona i/o institució que se'n va encarregar del seu desenvolupament i això és un greu problema per a les possibles modificacions que es puguin voler fer. A part d'això, es vol donar la possibilitat a l'usuari de fer servir una base de dades relacional. Els objectius que es volen assolir són:

- Dissenyar nova base de dades SQL
- Implementar la solució
- Poder configurar a quina BD guardar les dades

1.3 Estat de l'art

Actualment la plataforma RISCOSS, és un software que està en funcionament però no té manteniment des de l'octubre de 2016. Com a qualsevol programa, es necessita incorporar millores, eliminar bugs i afegir funcionalitats per tal que no quedi obsolet. Les millores que es poden fer són: estudi i canvi de components externs (respectant la llicència OSS del projecte), canvi en el front-end de la web, eliminació de bugs i canvi en la base de dades.

En aquest projecte, ens focalitzarem en canviar la base de dades. Actualment, la base de dades que s'utilitza per a guardar tota la informació de la plataforma és OrientDB[1]. OrientDB és una base de dades NoSQL open source de tipus graf. Aquesta base de dades està escrita en Java i es connecta a través d'una API per a realitzar les connexions. Els motius principals pels quals es volen fer els canvis és perquè no hi ha documentació del disseny ni de com funciona internament aquesta base de dades, no es treu profit dels avantatges d'una base de dades no relacional (l'escalabilitat) i el desenvolupament amb bases de dades SQL és més comú entre els desenvolupadors. Per tal d'optimitzar el temps, s'han considerat bases de dades gratuïtes amb les quals la desenvolupadora hagi treballat abans (PostgreSQL[2] i SQLite[3]).

PostgreSQL. És fàcil d'instal·lar, és escalable, permet guardar objectes JSON i té suport en línia.

SQLite. SQLite és una base de dades relacional on les transaccions són ACID (Atomicity, Consistency, Isolation, Durability), hi ha suport disponible, però no és recomanada per a aplicacions Client/Servidor i webs amb un gran volum. Com que no es recomana per a aplicacions client/Servidor, no podem escollir-ne aquesta opció.

Per concloure aquest apartat, no es pot aprofitar cap altre projecte existent perquè s'ha d'implementar una solució a mida sense canviar el funcionament actual amb el disseny específic de bases de dades de RISCOSS. Per dissenyar la nova part, s'utilitzarà PostgreSQL perquè és la més adequada i la més familiar per a la desenvolupadora, com s'ha dit anteriorment.

[1] "OrientDB." https://orientdb.com/. Accedit 20 Set. 2017.
[2] "PostgreSQL." https://www.postgresql.org/. Accedit 23 Set. 2017.
[3] "SQLite." https://www.sqlite.org/. Accedit 23 Set. 2017.

2. Abast del projecte

L'objectiu d'aquest projecte és fer reenginyeria de la plataforma que ja està en funcionament. La desenvolupadora i la codirectora junt amb el director es van posar d'acord per tal de focalitzar la millora a la base de dades perquè a causa del temps de duració del TFG no es poden realitzar totes. En voler fer la millora esmentada, també es rectificarà la documentació i estructuració del codi.

Per a poder realitzar els canvis, hi haurà tres etapes diferents que seguiran l'ordre següent:

- **Anàlisis de l'arquitectura i tecnologies actual**
  El primer pas és estudiar les eines que s'utilitzaran i familiaritzar-se amb el projecte. Les eines que s'utilitzaran ja són conegudes pel programador, tant el llenguatge com l'entorn i aquest procés serà més curt. D'altra banda, amb la documentació que hi ha del projecte i el codi, la desenvolupadora haurà d'entendre com està estructurat el projecte i si la documentació plasma el que hi ha a la realitat.
- **Identificació d'aspectes a millorar**
  A causa que el TFG té un temps acotat, no hi ha temps per migrat tota la informació que actualment es guarda a la base de dades, motiu pel qual hem decidit incloure només la part relacionada amb la gestió d'usuaris.
- **Implementació, proves, integració i documentació de la millora**
  Un cop escollit l'aspecte a millorar, es farà el disseny, s'implementarà i finalment es faran les proves per garantir el bon funcionament de la millora feta. A més a més, s'integrarà la millora al projecte i es deixarà tot documentat perquè els futurs professionals que hagin de fer modificacions puguin trobar tota la informació en un mateix lloc.

2.1 Possibles obstacles

2.1.1 Mal estat del codi

En aplicacions de software de dimensions grans on han participat diverses persones és comú que hi hagi parts que no s'hagin dissenyat de forma correcta per diversos motius com ara falta de temps o errors en el disseny.

La implementació de RISCOSS s'ha fet en mòduls separats i es podria donar el cas que el codi que implementa la base de dades no estigui centralitzat i sigui més complicat fer el canvi. Per poder solucionar-ho, s'estudiaran les diferents possibilitats de millores que es poden fer en el software. En cas que no es pugui fer perquè faltaria temps, el que es faria és tot l'estudi hipotètic i teòric en comptes de fer una implementació, tal com s'ha dit anteriorment.

2.1.2 Mal disseny

El disseny és una de les parts més importants d'un projecte i si es fa de manera errònia, es pot acabar fent feina inservible a més a més de perdre temps. En aquest projecte es podria fer una mala elecció del canvi a fer i una mala elecció del disseny a fer. Per evitar problemes de disseny, tota la feina que es faci es mostrarà a la codirectora a les reunions periòdiques per tal de saber si s'ha realitzat un disseny correcte.

2.1.3 Errors en el codi

En qualsevol desenvolupament de software quan s'han de fer modificacions de codi o afegir-ne de nous és molt probable que s'introdueixin errades. Per evitar possibles errades s'ha de provar al màxim els canvis fets i tenir en compte totes les possibilitats que hi poden haver. Per a poder solucionar-ho, s'han de fer jocs de proves per a garantir el correcte funcionament. A més a més, s'encapsularà tot el nou codi en un nou mòdul.

2.1.4 Integració dels canvis al projecte

Un cop es facin tots els canvis i fet les proves s'hauran d'incorporar tots els nous canvis fets al projecte. Els processos d'integració són crítics perquè el projecte en conjunt ha de funcionar igual que funcionava abans de fer res i ha de funcionar també la part inclosa. Per a poder solucionar-ho, s'hauran de fer proves de tots el casos crítics tant abans com després de la integració per assegurar que tot funciona tal com s'estava fent fins aquell moment.

2.1.5 Calendari

En tots els projectes hi ha dates límit per a fer l'entrega. En aquest cas, tenim un temps específic molt limitat en el que s'ha de fer el projecte i s'ha de tenir molt present perquè es pugui fer tota la feina que s'ha de fer.

Per a solucionar-ho, primer s'estudiarà quines són les possibles millores que es poden fer en aquest software i decidir quina es pot fer en aquest temps. A més a més, es faran reunions periòdiques per a poder redirigir el projecte el més ràpid possible en cas de necessitat.

2.2 Metodologia i rigor

A continuació es descriuran els mètodes de treball, les eines de seguiment i el mètode de validació utilitzats en el projecte.

2.2.1 Mètodes de treball

La metodologia de desenvolupament de software que s'utilitzarà en aquest projecte és el model en cascada perquè l'equip de desenvolupament és petit, no es necessiten cicles i no hi ha clients. Aquest model segueix un procés seqüencial que està dividit en diverses fases: Anàlisi de requeriments, disseny, implementació i proves, integració i proves i manteniment[4]. En tractar-se d'un projecte de reenginyeria, no tenim nous requisits, el que fa que a la fase d'anàlisi i disseny, el que analitzarem serà el codi actual per analitzar l'arquitectura actual de la plataforma. A continuació s'expliquen detalladament les fases que hi haurà i l'ordre en el qual es faran:

2.2.1.1 Anàlisi i Disseny

En aquesta primera fase es fa el disseny arquitectònic per a definir l'estructura actual de la plataforma, els mòduls que té i com es relacionen. D'aquesta manera s'estudiarà si és viable fer la millora de la base de dades o si s'ha de modificar el guió. El següent pas serà fer el disseny de la nova solució de la base de dades.

2.2.1.2 Implementació i proves

Seria la segona fase del projecte, on es farien la implementació i els tests per comprovar el correcte funcionament del codi desenvolupat. En aquesta fase també es faran proves per confirmar que la inclusió del nou codi no ha afectat la resta de funcionalitats.

2.2.1.3 Integració i manteniment

És l'última fase on es fa la integració de la millora i es fan actualitzacions del software per a introduir millores, solucionar bugs, afegir funcionalitats, ... Aquesta última fase no entrarà dins del projecte perquè no està acotat en el temps.

Paral·lelament, hi haurà una comunicació freqüent amb la codirectora del projecte. A la pràctica, això seran reunions setmanals per fer un seguiment detallat de l'evolució del projecte. D'aquesta manera, es podrà validar tota la feina feta i fer rectificacions com més aviat millor per a evitar problemes en la planificació. A més a més, aquestes reunions seran útils per a poder resoldre dubtes.

[4] Rentería, Miguel Ángel Sámano, and Ramón Rivera Espinosa. "Implementación del modelo integral colaborativo (MDSIC) como fuente de innovación para el desarrollo ágil de software en las empresas de la zona centro-occidente en México." (2014).http://www.eumed.net/tesis-doctorales/2014/jlcv/software.htm. Accedit 24 Set. 2017.

2.2.2 Eines de seguiment

Aquest projecte té com a objectiu fer reenginyeria i com que el codi de la plataforma ja està a GitHub, s'utilitzarà aquesta eina per a tenir control de versions i disponibilitat del codi. Per tal de garantir que el codi que està disponible al repositori funciona correctament, inicialment es plantejava l'opció de fer branques i/o fork per a fer el desenvolupament. Finalment, s'ha fet un fork i no s'han creat branques. S'ha decidit fer el fork perquè quan el nou mòdul funcioni es pugui integrar amb la resta del codi. D'altra banda, no s'han utilitzat branques perquè només hi ha hagut un desenvolupador i totes les modificacions estaven relacionades amb la nova base de dades. La wiki de GitHub[5] s'utilitzarà per actualitzar la documentació del projecte.

Per a la realització dels diagrames de Gantt s'utilitzarà el programa ProjectLibre[6] i per als diagrames Cacoo[7] i l'eina ObjectAid UML[8].

La documentació és una altra part molt important del TFG i per això, s'utilitzarà Google Drive[9]. D'aquesta manera evitarem la pèrdua de documents i la codirectora tindrà accés per a veure l'evolució del treball i fer les correccions que pertoquin remotament.

Com a eina de comunicació directa entre el desenvolupador, el director i codirectora s'utilitzarà el correu electrònic i com s'ha esmentat abans, hi haurà reunions setmanals.

2.2.3 Mètodes de validació

La planificació setmanal de reunions entre la desenvolupadora i la codirectora per a fer un seguiment continu de la feina feta (revisar el que s'ha fet, prendre decisions i corregir possibles errades) en conjunt amb el disseny de proves que es farà per a testejar les millores introduïdes en el sistema seran els mètodes de validació utilitzats. Per fer les proves, també s'utilitzaran casos de prova que es van utilitzar en la implementació de RISCOSS.

[5] "GitHub." https://github.com/. Accedit 30 Set. 2017.
[6] "Projectlibre." https://www.projectlibre.com/. Accedit 30 Set. 2017.
[7] "Cacoo." https://cacoo.com/. Accedit 30 Set. 2017.
[8] "ObjectAid." http://www.objectaid.com/. Accedit 30 Nov. 2017.
[9] "Google Drive" https://www.google.com/drive/. Accedit 30 Set. 2017.

3. Planificació temporal inicial

El projecte té una duració aproximada de 7 mesos, des de juliol 2017 fins al gener 2018. S'ha planificat la finalització del projecte a finals de desembre, l'última setmana, de manera que hi hagi temps de marge per poder actuar en cas de desviació i es pugui preparar la documentació i la defensa del TFG.

3.1 Descripció de les tasques i recursos

La feina que s'ha de fer en aquest projecte s'ha dividit en les següents fases:

3.1.1 Gestió de Projectes (GEP)

Aquesta primera fase correspon a l'assignatura semipresencial GEP on es fan les entregues que s'indiquen a continuació:

- Abast del projecte i contextualització
- Planificació temporal
- Gestió econòmica i sostenibilitat
- Presentació preliminar
- Plecs de condicions de les especialitats
- Presentació oral i document final

3.1.2 Anàlisi i disseny

En aquesta fase es fa l'anàlisi de l'arquitectura i tecnologies actuals, s'identifiquen els aspectes a millorar i es fa el disseny conceptual i el disseny de classes de millora de la base de dades.

3.1.3 Implementació i proves

En aquesta fase es fa tot el desenvolupament del projecte, és a dir, s'implementa tot el sistema seguint la documentació i decisions preses en la fase anterior. També és on es fan les proves de la part implementada.

3.1.4 Documentació

En aquesta última fase es fa la memòria final del TFG i es prepara la defensa oral.

3.2 Diagrama de Gantt inicial

A la secció *“14.1 Diagrama de Gantt inicial”* es troba el diagrama inicial, on es mostra el nom de la tasca, quan s'inicia i quan finalitza. Com es pot observar, la fase d'anàlisi i disseny és llarga perquè es tracta d'una reenginyeria on aquest procés és llarg.

3.2.1 Distribució estimada de l’esforç inicial

Fase	Duració (hores)
Gestió de Projectes (GEP)	75
Anàlisi i disseny	135
Implementació i proves	190
Documentació	50
TOTAL	450

Taula 1: Estimació en hores de les fases del projecte

3.3 Recursos

3.3.1 Recursos humans

En aquest projecte, només hi haurà un treballador com a recurs humà. Aquesta persona farà la funció de diferents rols per tal de fer totes les tasques necessàries en el TFG, a continuació s'indiquen cadascun d'aquests:

- **Cap de projecte**: Gestiona equips compostos per analistes, programadors, etc. També és el màxim responsable de la planificació i l'execució del projecte dins dels terminis establerts amb el client final, segons els estàndards de qualitat definits, i dins d'un marge de costos determinat.
- **Analista**: S'encarrega d'estudiar els problemes actuals del software i proposar diferents solucions i documentar-ho tot.
- **Dissenyador**: A partir de la documentació de l'analista, el dissenyador és qui defineix l'arquitectura i l'estructura de les funcionalitats que s'han de desenvolupar.
- **Programador**: Persona encarregada d'implementar la millora aplicant el disseny fet prèviament pel dissenyador.

3.3.2 Recursos materials i software

A continuació, es mostren els materials i software necessaris per a poder fer el projecte:

- **Ordinador portàtil**: Serà l'eina hardware que s'utilitzarà per fer el projecte.
- **Eclipse**: Entorn de programació on es desenvoluparan les millores. És l'entorn que s'ha utilitzat fins al moment.
- **Git i GitHub**: Software de control de versions i repositori en línia, respectivament.
- **Google Drive**: Plataforma on es compartiran els arxius de documentació del TFG.
- **ProjectLibre**: Programa de software per a realitzar esquemes de Gantt.
- **Correu electrònic**: Eina de comunicació que s'utilitzarà per a contactar amb el director i la codirectora.
- **Cacoo:** Eina de software per la creació de diagrames de classes i conceptuals.
- **ObjectAid UML:** Eina per a Eclipse que mostra automàticament el diagrama de classes del projecte.

3.4 Valoració d’alternatives i pla d’acció

En la planificació del TFG s'ha de tenir en compte que hi pot haver imprevistos i que aquests poden ser que apareguin amb més probabilitat en fases concretes, com ara a la fase d'implementació. Per aquesta raó, la fase d'implementació és la que té més hores de dedicació a la planificació.

Per tal de tenir marge en el temps en cas de necessitat, s'ha acotat el desenvolupament del projecte fins al desembre.

En la següent taula es mostren els possibles obstacles que podem trobar en la realització del projecte:

Risc	Gravetat	Pla de contingència
Mal estat del codi	Mitja	En cas de trobar-se un mal disseny del codi, s'hauran de fer plans alternatius, és a dir, altres millores com ara la millora de les llicències o fer un estudi teòric de què és el que s'hauria de fer.
Poca experiència en tecnologies, eines de desenvolupament i llenguatges de programació nous	Mitja	Es consultarà informació a webs i es farà un estudi previ de l’arquitectura del projecte per tal de familiaritzar-se amb l’entorn.
Pèrdua de codi o documentació del TFG	Alta	Es podran recuperar les últimes versions de codi al repositori (GitHub) i les últimes modificacions realitzades als documents a Google Drive.
Comprensió estat actual del codi	Alta	S’utilitzarà la documentació disponible i en cas de necessitat, es contactarà amb desenvolupadors de la plataforma.
Mala planificació	Baixa	En les reunions setmanals amb la codirectora es podran corregir desviacions o problemes.

Taula 2: Valoració de riscos i pla de contingència

En cas que es necessiti més temps del que es disposa per poder finalitzar el desenvolupament del projecte el que es farà és incrementar el temps de dedicació per tal de finalitzar el TFG en un quadrimestre, al gener.

4. Gestió econòmica inicial

4.1 Estimació de costos

En aquest apartat es farà un desglossament del pressupost total del projecte tenint en compte els recursos que s'utilitzaran, els imprevistos i la contingència.

4.1.1 Recursos humans

El projecte el desenvoluparà una persona que tindrà diferents rols durant el projecte. En la següent taula es mostra per cada rol, el preu per hora que cobra cada un (s'ha consultat la remuneració que tenen els perfils tècnics segons la consultora de contractació Page Personnel[10]), les hores previstes que dedicarà al projecte i el cost total que tindrà.

Rol	Preu per hora (€/h)	Hores previstes	Cost estimat(€)
Cap de projecte	45,00	125	5.625,00
Analista	40,00	95	3.800,00
Dissenyador	35,00	40	1.400,00
Programador	25,00	190	4.750,00
TOTAL		450	15.575,00

Taula 3: Pressupost de recursos humans inicial

[10] "Estudio de remuneración 2017 - Michael Page." https://www.michaelpage.es/sites/michaelpage.es/files/MP_SPA_ON_ER_IT_03052017.pdf. Accedit 09 Oct. 2017.

A continuació, es mostra el cost desglossat per cada activitat que hi ha al projecte.

	Temps de dedicació (en hores)				
Fase	Cap de projecte	Analista	Dissenyador	Programador	Cost estimat (€)
Gestió de projectes	75	0	0	0	3.375,00
Abast del projecte i contextualització	15	0	0	0	675,00
Planificació temporal	11	0	0	0	495,00
Gestió econòmica i sostenibilitat	11	0	0	0	495,00
Presentació preliminar	10	0	0	0	450,00
Plecs de condicions de les especialitats	8	0	0	0	360,00
Presentació oral i document final	20	0	0	0	900,00
Anàlisi i disseny	0	95	40	20	5.700,00
Instal·lació entorn i projecte	0	0	0	20	500,00
Anàlisi arquitectura i tecnologies actuals	0	90	0	0	3.600,00
Identificació aspectes a millorar	0	5	0	0	200,00
Disseny conceptual/classes de la part de la BD	0	0	40	0	1.400,00
Implementació i proves	0	0	0	170	4.250,00
Desenvolupament codi	0	0	0	105	2.625,00
Proves	0	0	0	50	1.250,00
Integració a la resta del projecte	0	0	0	15	375,00
Documentació	50	0	0	0	2.250,00
Memòria final	40	0	0	0	1.800,00
Preparació defensa oral	10	0	0	0	450,00
TOTAL	125	95	40	190	15.575,00

Taula 4: Costos directes per activitat inicials

4.1.2 Recursos hardware

Per al desenvolupament del TFG només s'utilitzarà com a hardware un ordinador portàtil. Per deduir el cost d'amortització per hora s'ha establert que 1 any de vida útil són 250 dies laborables i que cada dia es treballa 4 hores diàries.

Producte	Preu (€)	Vida útil (anys)	Cost d'amortització (€/h)	Hores	Amortització (€)
Ordinador portàtil	898,00	4	0,2245	450	101,02
TOTAL					101,02

Taula 5: Pressupost de recursos hardware

Cost d'amortització (€/h) = Preu / (4 anys * (250 dies laborals * 4 hores/dia))

4.1.3 Recursos software

El software utilitzat per fer el desenvolupament del projecte és gratuït: Eclipse, Git/GitHub, ProjectLibre, Correu electrònic, Cacoo, ObjectAid UML, Google Drive. Microsoft Windows també és una versió gratuïta per a estudiants de la UPC.

4.1.4 Despeses generals inicials

En aquest apartat es mostren els costos indirectes associats al projecte. En aquestes despeses tindrem en compte el consum energètic que tindrà el portàtil i la connexió a Internet. A més a més, es comptabilitza la despesa del transport públic per les reunions periòdiques que es faran durant la durada del projecte.

Producte	Preu(€)	Període	Cost estimat(€)
Transport (T-Jove 1 zona)	35,00 €/mes	7 mesos	245,00
Connexió a Internet	30,00 €/mes	7 mesos	210,00
Consum energètic	0,14 €/KWh	450 hores	12,60
TOTAL			467,60

Taula 6: Pressupost de despeses generals

4.1.5 Imprevistos

L'imprevist que pot haver-hi durant la realització del projecte és l'avaria del portàtil. En aquest cas, s'hauria de reparar o substituir sent la segona la millor opció per a no perdre temps (Cost ordinador nou: 898,00 €). La probabilitat de tenir aquest imprevist és baixa perquè l'ordinador només té un any (se li assigna un 5%).

4.1.6 Contingència

Es reserva una part del pressupost per si hi ha desviacions en el cost previst del projecte. El nivell de contingència fixat és del 15% sobre el total de la suma dels costos directes i indirectes.

Producte	Preu(€)	Percentatge(%)	Cost(€)
Recursos humans	15.575	15	2.336,25
Recursos hardware	101,02	15	15,15
Despeses generals	467,60	15	70,14
TOTAL	16.143,62		2.421,54

Taula 7: Contingència

4.1.7 Pressupost total inicial

A continuació es mostren els conceptes i els costos resumits i el pressupost total del projecte:

Concepte	Cost estimat (€)
Recursos humans	15.575,00
Recursos hardware	101,02
Recursos software	0,00
Despeses generals	467,60
Imprevistos	44,90
Contingència	2.421,54
TOTAL	18.790,06

Taula 8: Pressupost total inicial

4.2 Control de gestió

El control de costos que es farà en aquest projecte majoritàriament és la dels recursos humans que és la que té relació directa amb el desenvolupament.

Al final de cada fase del projecte es calcularan les hores que s'han dedicat i es comprovarà si el total d'hores es correspon amb l'estimació del temps fet. En qualsevol cas el que s'ha d'evitar és que l'estimació es quedi curta perquè llavors hi hauria problemes de temps i diners.

Un cop acabat el projecte, en cas que el cost de desenvolupament sigui més gran que les estimacions fetes, s'haurà d'utilitzar la part de contingència per tal de cobrir les despeses.

El càlcul de les desviacions que es poden produir en el projecte es realitzarà amb les fórmules següents:

- Desviament recursos humans = (cost estimat - cost real) * hores reals
- Desviament en una fase del projecte = (cost estimat - cost real) * hores reals
- Desviament total de fases = cost total estimat de les fases - cost total real
- Desviament total en recursos = cost total estimat en recursos - cost total real en recursos

5. Informe de sostenibilitat

5.1 Dimensió econòmica

Tal com es mostra a l'apartat *"10.4.3 Pressupost total final"* s'han quantificat tots els costos necessaris per a dur a terme el projecte. Com es pot veure, és un pressupost viable econòmicament en el qual s'han utilitzat els recursos mínims necessaris. Per a poder reduir el pressupost s'hauria d'haver augmentat el número de desenvolupadors o haver-ho fet en menys temps, fet que resulta impossible a causa de la dedicació de temps a la universitat i a la feina que té la desenvolupadora. Com que és un pressupost viable econòmicament en el qual s'han utilitzat els recursos mínims necessaris, s'ha decidit posar-li un 9.

El cost que tindrà el projecte durant la seva vida útil és el salari dels desenvolupadors que seguiran contribuint al desenvolupament de la plataforma RISCOSS. Com que el cost serà el mateix que el dels recursos necessaris en la part de PPP, se li dóna un 18.

El risc econòmic que es podria donar és no tenir subvenció en investigació i com a conseqüència, no poder continuar amb el desenvolupament de la plataforma. Com que aquest risc és basant probable que passi a causa de les retallades en els pressupostos relacionats amb educació i investigació, se li dóna una puntuació de -15.

5.2 Dimensió social

En l'àmbit personal, la realització d'aquest projecte m'ha aportat coneixements en l'àmbit professional i tècnic. Aprendre a fer reenginyeria, i tot el que això comporta: haver d'entendre codi ja implementat, extreure el disseny i introduir noves línies de codi. A més a més, durant la realització del projecte m'ha fet reflexionar sobre la importància d'aplicar les bones pràctiques tant en l'àmbit de programació com de documentació per a facilitar els posteriors desenvolupaments. A causa d'aquest nivell d'impacte personal s'ha decidit posar-li un 10.

La solució donada en aquest projecte ajudarà als desenvolupadors, especialment als nous que segueixin amb la implementació de la nova BD, perquè s'ha fet el primer pas que és tota la configuració i estructuració perquè sigui fàcil la continuïtat.
A més a més, en ser un projecte Open Source el codi podrà ser utilitzat per qualsevol persona en altres projectes. D'altra banda, no hi ha cap col·lectiu que es pugui veure perjudicat pel projecte. Com que només s'ha pogut fer una petita part i no s'ha integrat amb la resta del projecte, se li dóna un 12.

En cap cas el projecte pot ser perjudicial per a la població, és a dir, no hi ha cap risc social i és per això que la puntuació de risc social és 0.

5.3 Dimensió ambiental

La realització del projecte s'ha aconseguit amb els recursos mínims necessaris. L'impacte final dels recursos utilitzats s'ha quantificat als apartats *"4.1.2 Recursos hardware"* i *"10.4.2 Despeses generals finals"*. En ser un projecte de reenginyeria és difícil saber quins recursos es van fer servir en total i saber si es podria reduir.

Els recursos fets servir per la plataforma seran exactament els mateixos. En cas que els canvis realitzats en aquest projecte s'integrin amb la resta del projecte, augmentarà l'ús de recursos per temes d'espai.

El risc que es pot donar és que la quantitat de dades sigui molt gran i es necessitin més recursos per a emmagatzemar-les.

La puntuació d'aquesta secció serà de 0 punts perquè no afecta ni positivament ni negativament al medi ambient.

5.4 Matriu de sostenibilitat

A continuació es mostra la matriu de sostenibilitat on es mostra, per cada un dels blocs, la valoració de l'impacte econòmic, social i ambiental per a la fita inicial. S'ha obtingut una puntuació de 34 respecte a (-60 : 90).

	PPP	Vida útil	Riscos
Ambiental	Consum del disseny	Petjada ecològica	Riscos ambientals
	0/10	0/20	0
Econòmic	Factura	Pla de viabilitat	Riscos econòmics
	9/10	18/20	-15
Social	Impacte Personal	Impacte Social	Riscos socials
	10/10	12/20	0
Rang de sostenibilitat	19/30	30/60	-15/-60
	34/(-60 : 90)		

Taula 9: Matriu sostenibilitat

6. Disseny del software

6.1 Introducció

En aquest apartat s'explica l'arquitectura de la plataforma RISCOSS, mostrada a la figura 1. La figura i la descripció dels components s'han extret del document "*D4.5-Specification and design of the RISCOSS platform"* del projecte RISCOSS, públic a la web del projecte[11].

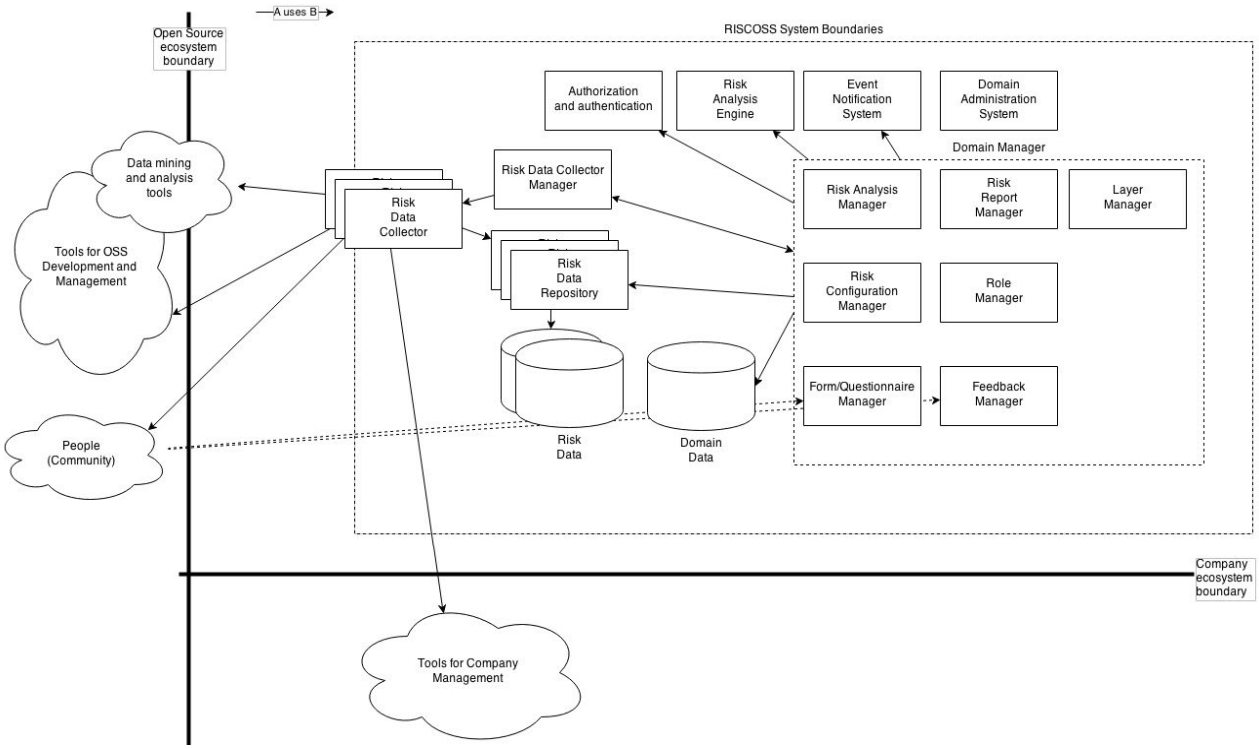


Figura 1: Arquitectura plataforma RISCOSS

A continuació es detallen els diferents components:

- **Risk Data Collector**: Components independents que són capaços de recollir dades sense processar de diferents fonts. La seva tasca és tractar aquestes dades perquè les pugui utilitzar la plataforma RISCOSS. Normalment, s'utilitzen per a recuperar dades automàticament que poden ser utilitzades com a input als anàlisis de risc d'una entitat.

[11] "riscoss.eu: Managing Risk and Cost in Open Source Software Projects" http://www.riscoss.eu/. Accedit 28 Jul. 2017.

- **Risk Data Collector Manager**: Aquest component gestiona els Risk Data Collectors disponibles en la plataforma donant, a través d'una API, informació sobre que està disponible i quins paràmetres es necessiten. També gestiona l'execució dels Risk Data Collectors.
- **Risk Data Repository**: Aquest component proporciona emmagatzematge i queries per a guardar i obtenir Risk Data. El Risk Data Collector Manager és el responsable de guardar les dades que són enviades dels Risk Data Collectors que gestiona.
- **Domain Administration System**: Aquest component és l'element central de la plataforma RISCOSS i dóna aïllament de les activitats de la plataforma RISCOSS. Un domini és un contenidor amb totes les dades d'un context. Una entitat que utilitzi RISCOSS, tindrà el seu domini aïllat dels altres. A cada domini, els usuaris poden definir els seus models, capes, rols, i guardar dades rellevants per a fer anàlisis de risc. Cada domini té les seves regles d'accés.
  Els dominis permeten diferents escenaris de desplegament:
  - Single domain deployment**.** Aquest seria el cas en el qual l'organització vol usar RISCOSS internament. Serà gestionat com qualsevol altre sistema informàtic intern.
  - Multiple domains deployment in a single organization. Aquest cas és similar a l'anterior, exceptuant que una organització pot voler tenir diversos dominis aïllats per a fer operacions.
  - Public single domain deployment**.** En aquest cas, l'organització vol utilitzar RISCOSS per a donar informació sobre els projectes que estan gestionant.
  - RISCOSS-As-A-Service. En aquest cas, l'actor decideix crear un negoci utilitzant la plataforma RISCOSS i proporcionar el hosting del domini a organitzacions disposades a pagar.
- **Authorization and authentication**: Aquest component proporciona funcionalitats d'autorització i autentificació per a la plataforma RISCOSS. El Domain Manager ho utilitzarà per a donar accés als recursos guardats en un domini donat un usuari.
- **Event Notification System**: Aquest component proporciona funcionalitats per al seguiment d'activitats i enviar notificacions quan hi ha esdeveniments determinats.
- **Risk Analysis Engine**: Aquest component conté tota la lògica per a fer anàlisis de risc fent servir les dades disponibles en un domini.
- **Domain Manager**: És una macro component que gestiona totes les dades que poden ser manipulades en un domini. Conté diversos subcomponents:
  - Layer Manager: Aquest component dóna les funcionalitats necessàries per definir i gestionar quines capes estan en un domini. Permet crear jerarquia de capes i definir l'estructura d'una entitat.
  - Role Manager: Aquest component dóna les funcionalitats necessàries per definir i gestionar quins rols estan actualment a un domini. Un rol defineix tipus d'usuaris i permet definir normes per a accedir a les dades i funcionalitats. Els tipus de rols que hi ha a la plataforma són:

    - Sysadmin: Aquest rol és l'encarregat de gestionar els aspectes administratius de la plataforma com per exemple gestionar contes d'usuari, permisos d'usuari o configurar quines extensions de la plataforma com ara instal·lar/configurar els Risk Data Collectors estan disponibles i visibles a través de la plataforma.
    - Producer: Aquest rol crea/manté les entitats de la plataforma.
    - Modeler: Aquest rol és l'encarregat de crear configuracions de risc. És la persona que entén que fan els models i la persona que els pot escriure.
    - Consumer: Aquest rol és el qui vol realitzar anàlisis de risc en entitats utilitzant les configuracions de risc disponibles.
- **Risk Analysis Manager**: Aquest component permet definir i fer sessions d'anàlisi de risc a una entitat d'un domini. També permet guardar els resultats dels anàlisis de risc.
- **Risk Configuration Manager**: Aquest component proporciona els mitjans per gestionar tots els artefactes necessaris per a un anàlisi de risc.
- **Model Manager**: Aquest component serveix per a gestionar el repositori de models que serà utilitzat per a fer les configuracions de risc que s'utilitzaran als anàlisis de risc. S'ha afegit aquest component per a poder simplificar la gestió de models que poden ser reutilitzats a les configuracions de risc.
- **Form/Questionnaire Manager**: Aquest component proporciona els mitjans per definir i gestionar els formularis d'usuari i inserir informació necessària per al Risk Analysis Engine.
- **Feedback Manager**: Aquest component proporciona els mitjans per gestionar el feedback dels usuaris de la gestió de dades en un domini. El feedback s'utilitza per a millorar els models utilitzats pel Risk Analysis Engine per a realitzar els anàlisis de risc.

6.2 Patrons de disseny

En aquest apartat es descriuen els patrons utilitzats en la nova implementació realitzada en aquest TFG.

6.2.1 Patró adaptador

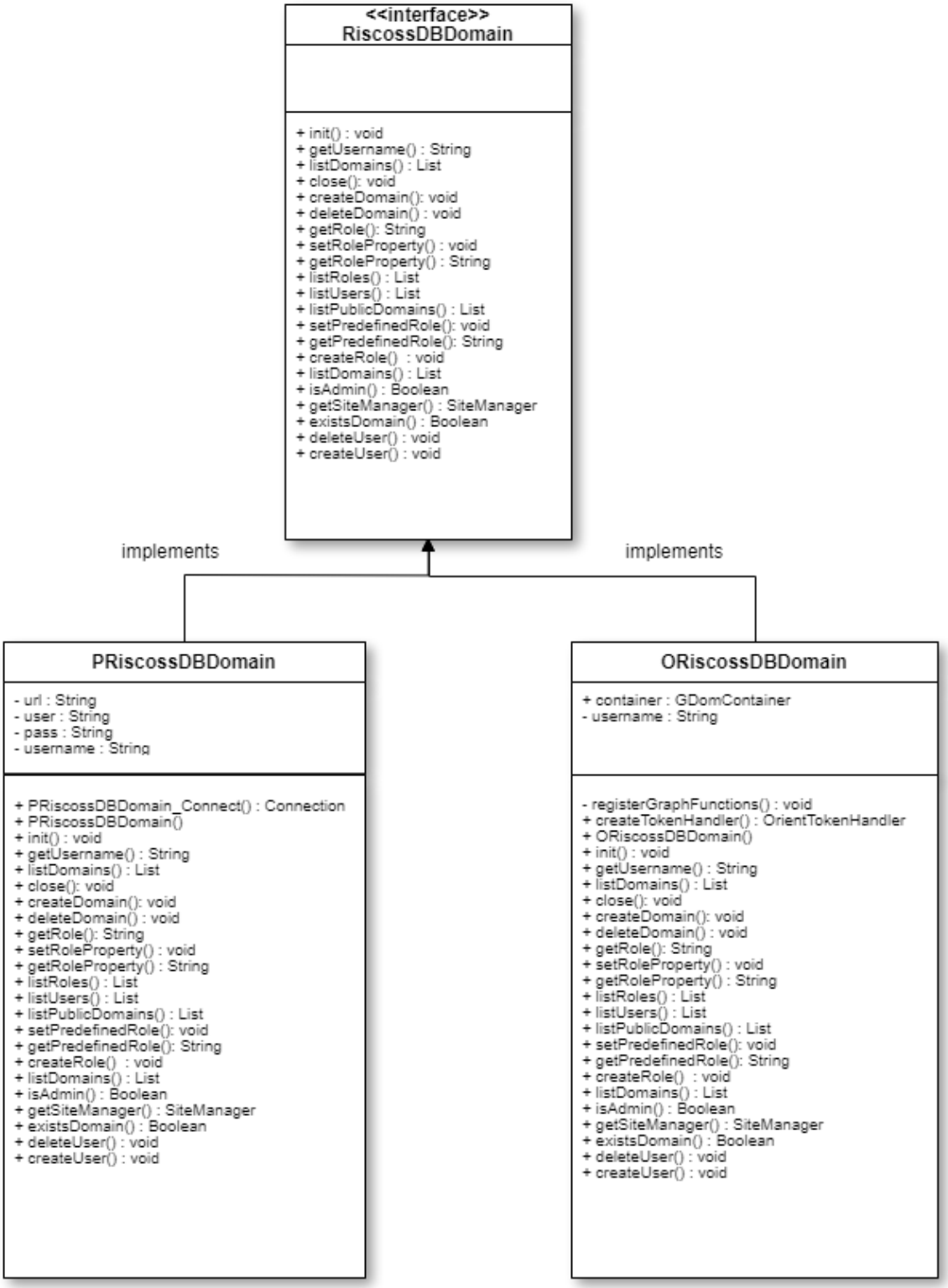


Figura 2: Patró adaptador accés a base de dades

Aquest patró es fa servir per desacoblar les capes de domini i dades, en concret la implementació de les classes que accedeixen a la base de dades.
L'aplicació del patró consisteix, per cada classe de la capa de dades, en implementar una interfície que és la que es crida des de les classes de domini i la classe que implementa l'accés a la base de dades.

En la versió actual, els programadors havien desenvolupat la interfície i la classe d'accés a la base de dades OrientDB. Per afegir l'accés a la base de dades PostgreSQL, s'ha implementat una segona classe implementant la interfície. El resultat ha estat, per cada classe de la capa de dades tenim: 1 interfície + 2 classes que implementen (OrientDB i PostgreSQL), tal com es mostra a la figura 2.

6.2.2 Patró Singleton

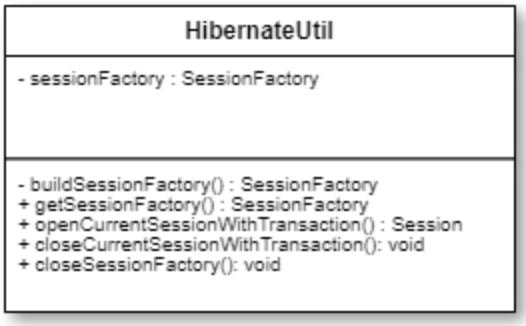


Figura 3: Patró singleton classe HibernateUtil

El patró Singleton es fa servir per a restringir la instanciació d'una classe a un objecte. Consisteix en una classe que ha de crear un objecte assegurant que es crearà només un objecte. Aquesta classe proveeix una manera d'accedir a l'objecte que pot ser accedit directament sense necessitar instanciar l'objecte de la classe.

El que s'ha fet és crear la classe HibernateUtil per a restringir l'accés a SessionFactory, que és un objecte on es guarda tota la configuració i mapeig de les dades i proporciona instàncies de les Hibernate Sessions amb les quals es fan les connexions a la BD. Com que aquest projecte és una aplicació web, hem de fer que no es creïn diversos objectes per tal de millorar el rendiment de l'aplicació. Com podem veure a la figura 3, hi ha una instància privada del SessionFactory i el seu constructor buildSessionFactory() també és privat. Per a accedir a la instància es pot utilitzar el mètode gestSessionFactory().

6.2.3 Patró DAO (Data Access Object)

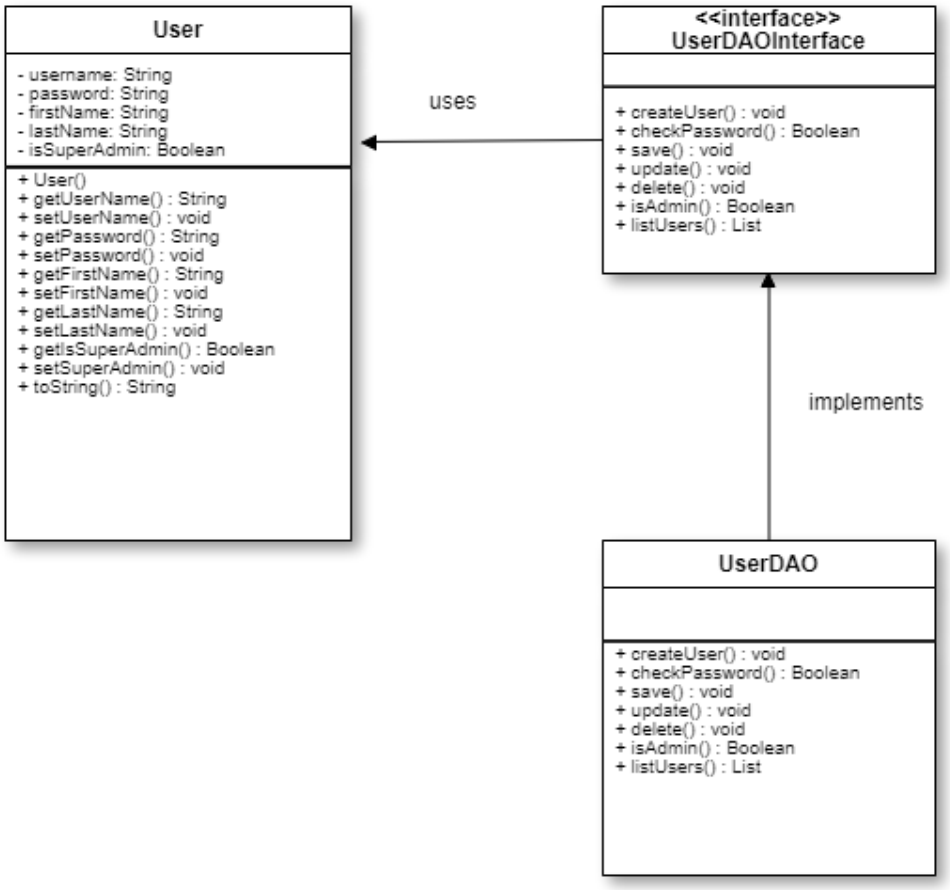


Figura 4: Patró DAO

Aquest patró es fa servir per aïllar l'aplicació de la persistència. Consisteix en una interfície que defineix les operacions que es realitzaran per a l'objecte en qüestió, una classe que implementarà les operacions definides a la interfície i un model que conté els getters i setters per a guardar les dades utilitzant la classe UserDAO.
El que s'ha fet és, per a cada objecte, un model amb la seva respectiva interfície i implementació. A la figura 4, podem veure com a exemple el disseny de com s'ha fet per a l'User.

6.3 Diagrama Conceptual

Després de l'anàlisi de la documentació i el codi, la figura 5 descriu el model conceptual de la informació que es fa servir a la plataforma RISCOSS.

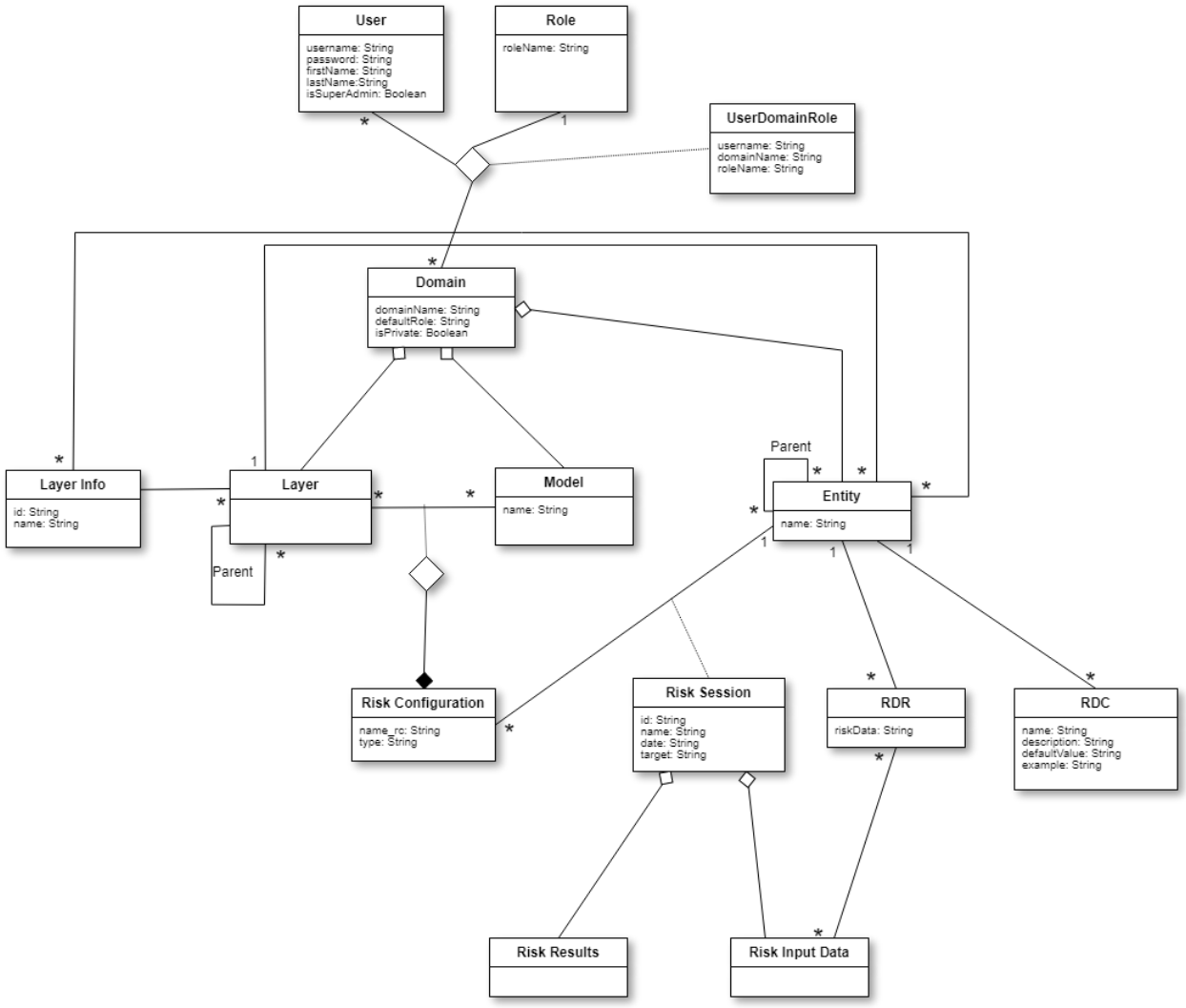


Figura 5: Model Conceptual de RISCOSS

Els conceptes es poden agrupar en:

- Gestió de dominis (Domain): El sistema guarda separadament tota la informació en diferents dominis, per poder mantenir la informació dels diferents usuaris independent.
- Gestió d'usuaris (User, Role, UserDomainRole): Els usuaris es poden assignar als dominis amb un rol corresponent. Els rols donen accés a les diferents funcionalitats de l'aplicació.
- Configuració de les entitats sobre les quals es farà l'anàlisi (Entity), organitzats en nivells (Layer i Layer Info)

- Configuració de l'anàlisi de risc: per poder analitzar els projectes OSS cal crear una configuració de risc (Risk Configuration) escollint un conjunt de models (Model) pels diferents nivells.
- Anàlisi de riscs: Cada vegada que es fa una anàlisi es guarda la informació de la sessió d'anàlisi (Risk Session), juntament amb les dades que es fan servir d'entrada (Risk Input Data), que és una còpia de les dades que hi ha al Risk Data Repository (RDR), i els resultats (Risk Results).

Donada la duració del projecte, s'ha d'escollir part del model per poder aplicar la persistència a la nova base de dades relacional. S'ha escollit els conceptes relacionats amb la gestió d'usuaris.

Les raons per escollir aquestes classes han estat:

- la quantitat de classes és adient per poder fer la implementació de totes les classes relacionades donada la duració del projecte
- la quantitat d'informació que es pot generar pels inputs i els resultats de les sessions de riscs pot ser considerable, motiu pel qual una base de dades NoSQL (com la OrientDB) pot ser més adequada.

6.4 Disseny nova BD

En aquesta secció es mostra el disseny de la part de la base de dades que s'ha implementat.

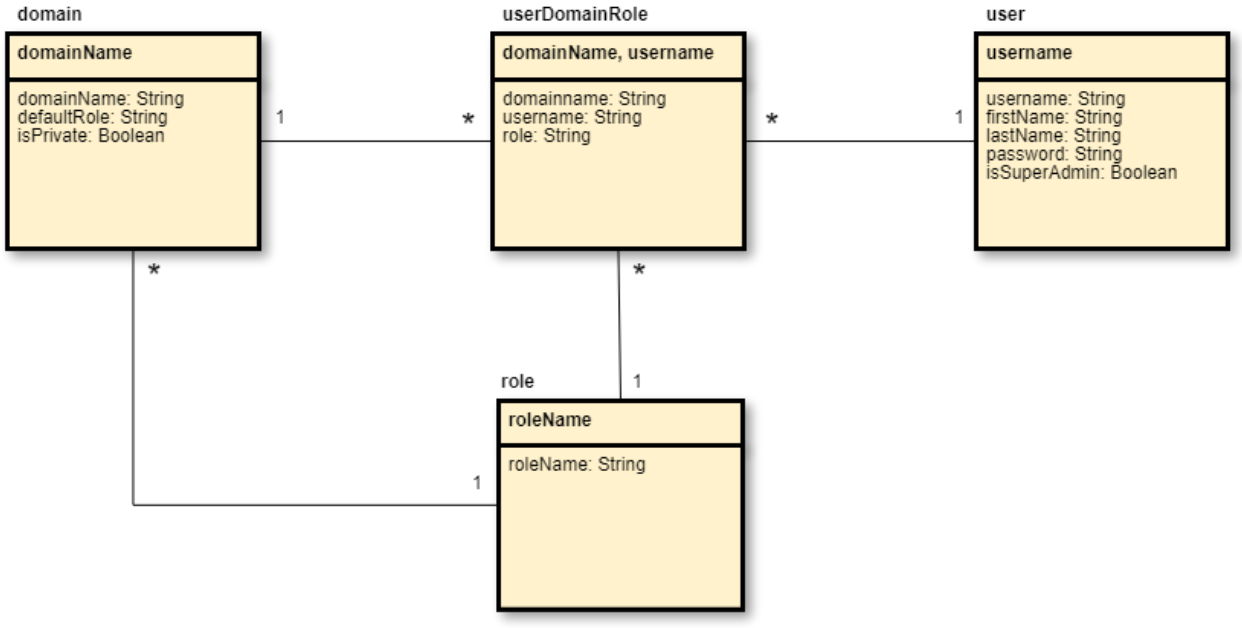


Figura 6: Model base de dades

6.5 Diagrama de classes

En aquest apartat es mostra el diagrama de classes del package postgreSQL relacionat amb les classes de la gestió d'usuaris.

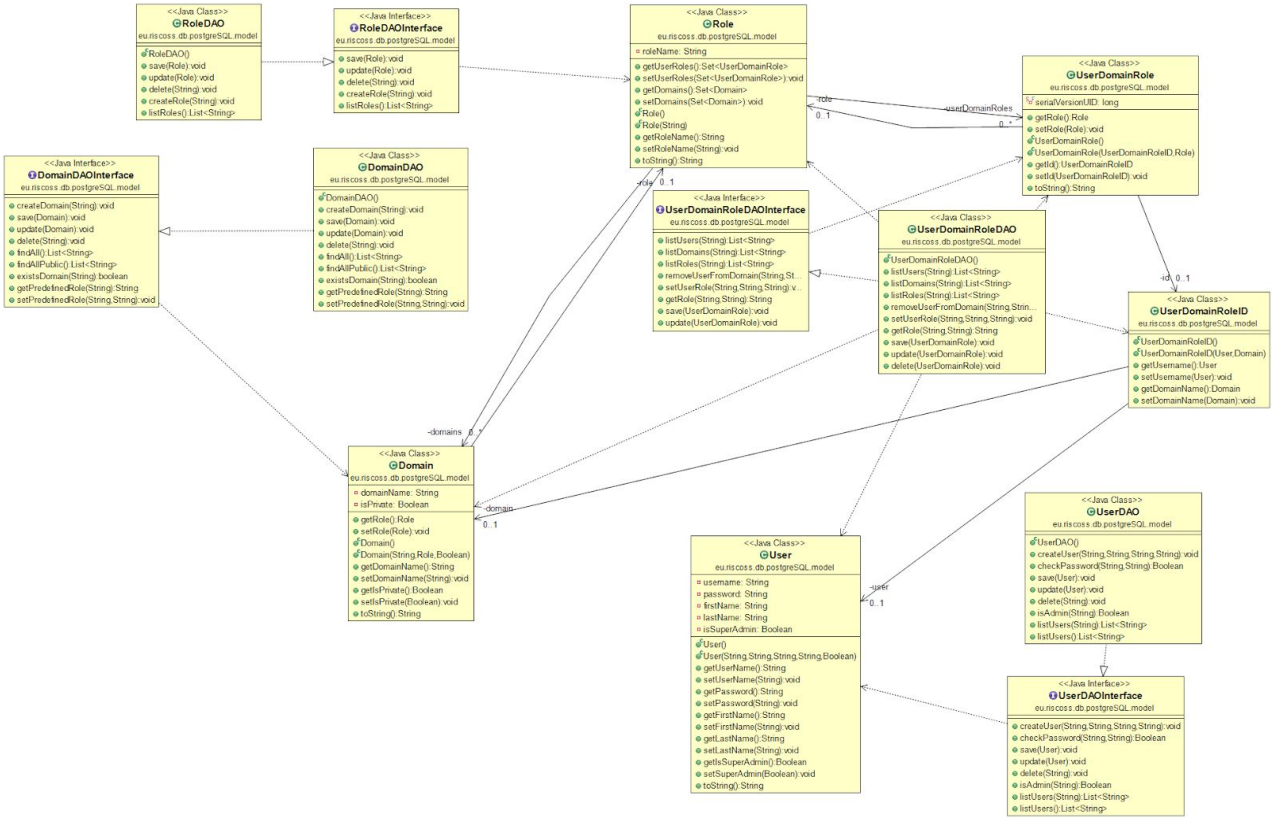


Figura 7: Diagrama de classes

6.6 Model de comportament

En aquest apartat es descriu el model de comportament que hi havia a la plataforma perquè no s'ha modificat cap comportament del sistema, és a dir, no hi ha hagut modificacions respecte al que hi havia a l'inici. La descripció del model de comportament ha sigut una part necessària per entendre el funcionament del sistema. Addicionalment, aquest anàlisi també ens ha ajudat a identificar les classes que gestionen l'accés a la base de dades, concretament identificant el patró adaptador, facilitant la feina de canvi de la base de dades.

6.6.1 Diagrames de seqüència

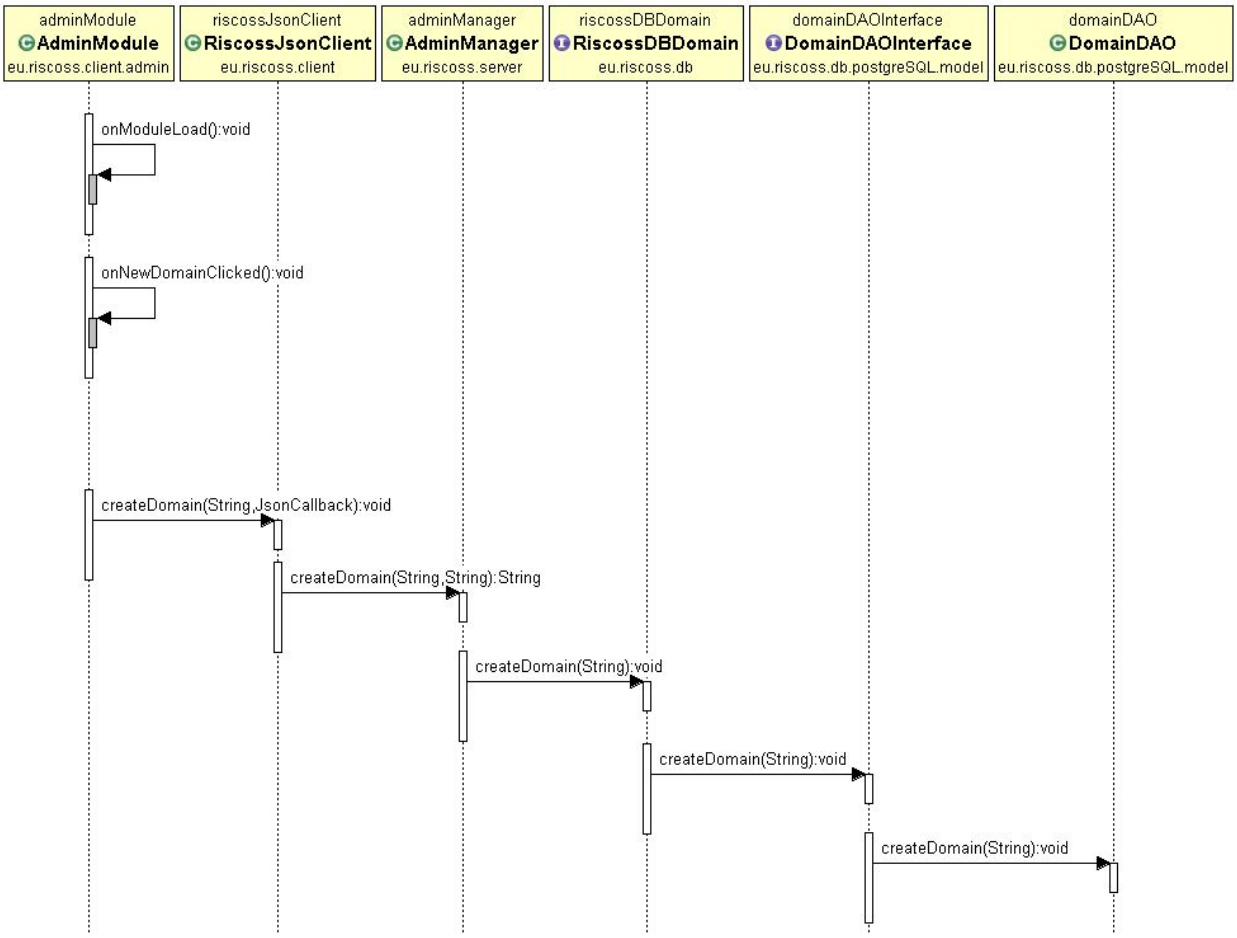


Figura 8: Diagrama de seqüència createDomain implementació PostgreSQL

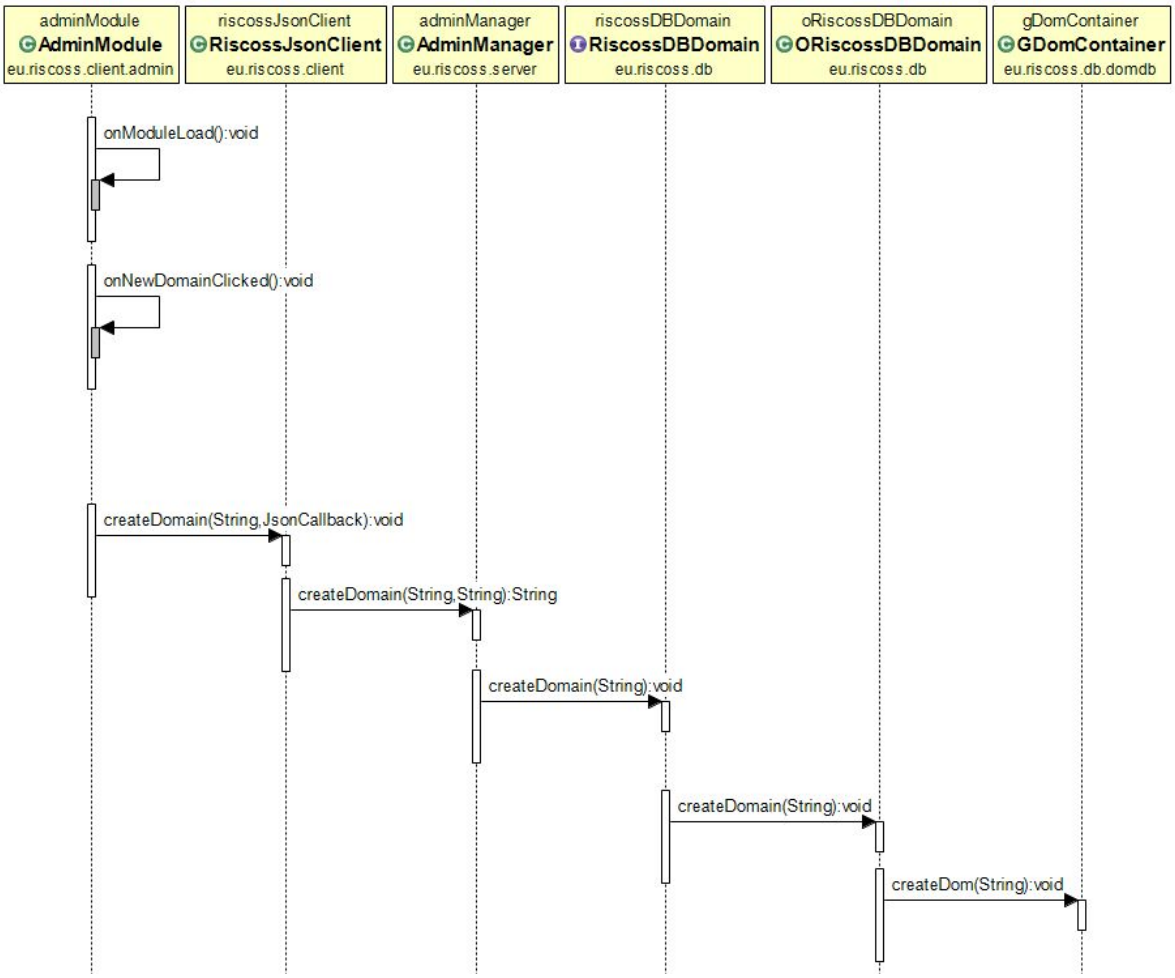


Figura 9: Diagrama de seqüència createDomain implementació OrientDB

Per tal de mostrar com són els diagrames de seqüència i com està estructurada l'aplicació, s'ha decidit crear el diagrama de seqüència de la funció createDomain com a mostra de com s'accedeix a la base de dades. A la figura 8 es mostra com s'accedeix amb la nova implementació de la part de PostgreSQL, i a la figura 9 es mostren les classes a les quals s'accedeix amb la implementació de la part d'OrientDB.

6.7 Requisits no funcionals

A causa que en aquest projecte s'ha fet reenginyeria, no s'han afegit nous requisits no funcionals. No hi ha documentació de quins requisits no funcionals s'han aplicat en el desenvolupament inicial de la plataforma RISCOSS, però s'han pogut fer deduccions de quins són els possibles requisits no funcionals aplicats. Per tal de poder identificar els requisits no funcionals a l'aplicació s'ha consultat l'estandard ISO/IEC 25010[12].

El requisit no funcional que s'ha aplicat per a la base de dades no SQL que s'utilitza actualment a la plataforma sembla que ha sigut l'escalabilitat, més concretament l'eficiència. Com que s'han de guardar una quantitat de dades molt gran a causa de la quantitat de dades processades, a l'inici es va tenir en compte i sembla que es va decidir una base de dades no SQL per a la gestió d'aquestes.

Respecte a l'arquitectura sembla que s'ha tingut en compte el requisit de mantenibilitat, concretament la modularitat i la reusabilitat. S'ha arribat a aquesta conclusió perquè el codi està dividit en diferents mòduls, tal com s'ha mostrat en apartats anteriors, amb l'aparent intenció de què els diferents components siguin independents i tinguin el menor impacte possible en els altres components. Relacionat amb això, aquesta manera d'estructurar el codi fa que sembli que la idea inicial era que els mòduls poden ser reutilitzats en altres projectes sense afectar el funcionament de la plataforma.

L'altre requisit no funcional a destacar és el de seguretat, concretament la confidencialitat i l'autenticitat. Aquest requisit no funcional està estretament relacionat amb com es guarda la informació d'usuari i contrasenyes perquè per fer login s'ha d'introduir el nom d'usuari i contrasenyes definides anteriorment per l'usuari i la contrasenya s'encripta.

[12] "ISO 25010 - ISO/IEC 25000." http://iso25000.com/index.php/en/iso-25000-standards/iso-25010. Accedit 20 Mar. 2018.

7. Implementació

7.1 Introducció

El codi on es pot trobar als repositoris oberts de la plataforma RISCOSS que s'indiquen a continuació:

- Repositori RISCOSS: https://github.com/RISCOSS
- Repositori riscoss-corporate: https://github.com/RISCOSS/riscoss-corporate
- Fork repositori riscoss-corporate: https://github.com/Nalaboo/riscoss-corporate

El projecte RISCOSS està dividit en diferents repositoris els quals tenen diferents objectius. Concretament, els repositoris són:

- **Riscoss-corporate, riscoss-githubanalyser, riscoss-community**: són tres diferents front-ends que té la plataforma.
- **Riscos-data-collector**: es troben els components que llegeixen les dades dels projectes OSS (Open Source Software) per fer-les servir com a input de l'anàlisi de riscos.
- **Riscoss-risk-modeling**: on es comparteixen els diferents models que es fan servir per configurar l'anàlisi de riscos.
- **Riscoss-analyser**: conté la lògica per a realitzar els anàlisis utilitzant les dades d'un domini donat

L'àmbit d'aquest projecte és el repositori riscoss-corporate, a les següents subseccions es donen els detalls del projecte que es troba en aquest repositori.

7.2 Informació sobre els components del sistema

7.2.1 Tecnologies

Maven

Apache Maven[13] és una eina de gestió i construcció de projectes software. El que utilitza aquesta eina és un POM (Project Object Model) basat en XML per a descriure el projecte de software amb les dependències amb altres mòduls i components externs.

[13] "Maven – Welcome to Apache Maven." https://maven.apache.org/. Accedit 04 Mar. 2018.

GWT (Google Web Toolkit)
GWT[14] és un entorn de treball per a crear aplicacions web escalables i amb alt rendiment fent que siguin fàcils de mantenir, és a dir, crear RIAs (Rich Internet Application). Una RIA és una aplicació web que té la majoria de característiques que tenen les aplicacions d'escriptori tradicionals. Alguna de les seves característiques més importants són:

- Compila el codi escrit en Java a JavaScript
- GWT crea automàticament codi JavaScript compatible amb cada navegador.
- Fàcil integració amb Maven
- És open source i té la llicència Apache versió 2.0

Concretament, s'ha utilitzat el plugin per a Eclipse.

Llicència Apache 2.0(AP2.0)
La llicència Apache[15] és una llicència de programari lliure permissiva que requereix la conservació de l'avís de copyright. Es permet plena llibertat per a l'ús, distribució, modificació i distribució de versions modificades del programari. El que cal afegir al programa són dos arxius, un amb la còpia de la llicència i un altre document de text amb els avisos obligatoris del programari.

Hibernate
Hibernate[16] és una eina ORM (Object-Relational Mapping) implementada per aplicacions Java. Proporciona un entorn de treball per a mapejar un domini a una base de dades relacional. És programari lliure i es va tenir contacte amb aquesta eina a l'assignatura AS (Arquitectura de Software). La raó per la qual s'ha escollit utilitzar aquesta eina és per a facilitar el posterior desenvolupament, ja que és un model molt complex i també per a donar la possibilitat que si es vol canviar la base de dades es pugui fer afectant el menys possible al desenvolupament fet.

PostgreSQL
PostgreSQL[17] és una base de dades relacional open source. La raó per la qual s'ha escollit aquesta BD és perquè s'utilitza en l'assignatura de Bases de Dades i la desenvolupadora ja l'ha utilitzat.

[14] "[GWT]." http://www.gwtproject.org/. Accedit 04 Mar. 2018.
[15] "Apache License, Version 2.0 - The Apache Software Foundation!." https://www.apache.org/licenses/LICENSE-2.0. Accedit 04 Mar. 2018.
[16] "Hibernate. Everything data. - Hibernate." http://hibernate.org/. Accedit 04 Mar. 2018.
[17] "PostgreSQL." https://www.postgresql.org/. Accedit 04 Mar. 2018.

7.2.2 Estructura del projecte

En aquest apartat es mostren els packages que formen riscoss-corporate i també quins són els packages que s'han hagut de modificar i crear per tal de fer la implementació.

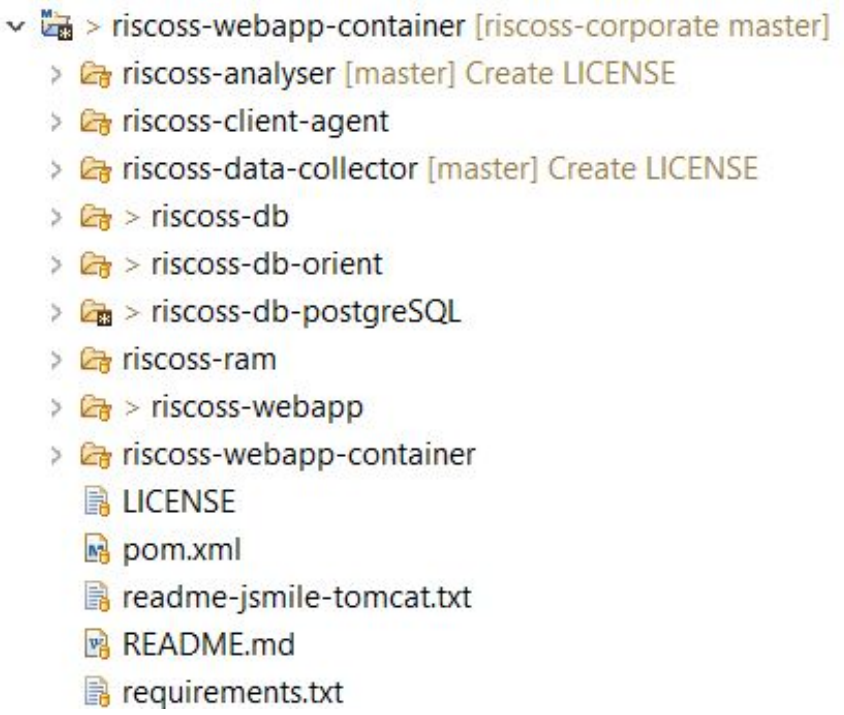


Figura 10: Packages del riscoss-corporate

Com podem observar a la figura 10, riscoss-corporate està compost de diferents packages que s'encarreguen de les diferents funcionalitats. Els packages són:

- **Riscoss-analyser** conté la lògica per a realitzar els anàlisis utilitzant les dades d'un domini donat i com s'ha vist abans, està separat en un altre repositori.
- **Riscoss-client-agent** conté les crides dels RESTServices de l'aplicació, com per exemple operacions per a obtenir el rol d'un domini o eliminar un usuari.
- **Riscoss-data-collector** conté els components que llegeixen les dades dels projectes OSS per fer-les servir com a input de l'anàlisi de riscos. Aquesta part és un altre repositori a part dels que hi ha al repositori RISCOSS.
- **Riscoss-ram** conté la implementació relacionada amb els anàlisis de risc.
- **Riscoss-webapp** conté el front-end i la implementació de les funcions REST (relacionat amb el package riscoss-client-agent).
- **Riscoss-webapp-container** Carpeta que es crea amb el resultat de si s'ha generat o no el .war.
- **Riscoss-db** que es veurà amb més detall a continuació, conté les interfícies de la base de dades.

- **Riscoss-db-orient** relacionat amb la BD tenim també aquest package que és on està la implementació de totes les funcions definides a les interfícies del package riscoss-db.
- **Riscoss-db-postgreSQL** que és el nou package i conté la nova implementació.

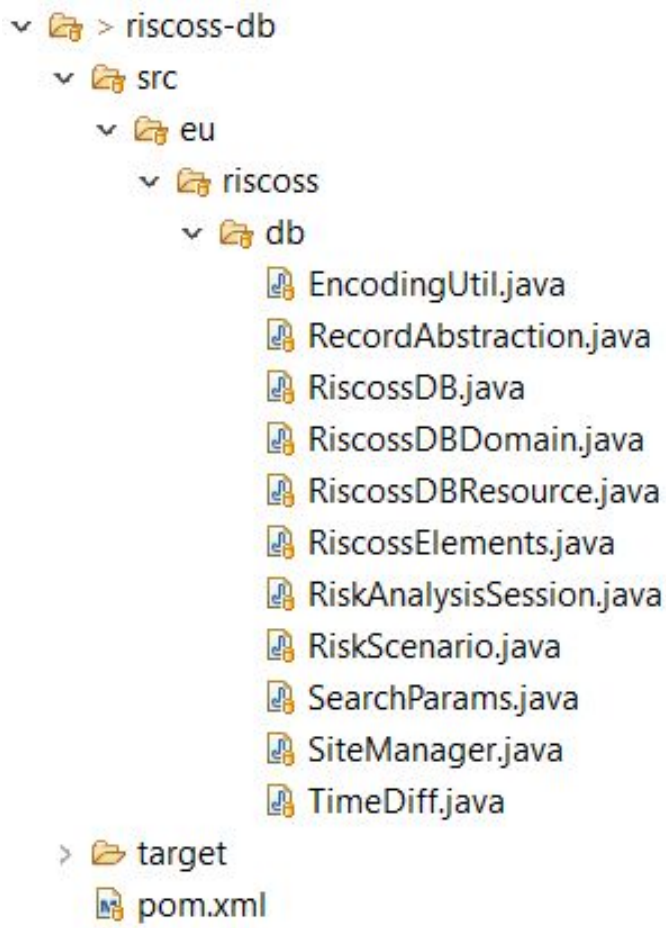


Figura 11: Package riscoss-db

En el package riscoss-db, tal com s'ha esmentat abans, trobem les interfícies relacionades amb la base de dades. El que s'ha canviat són alguns fitxers de lloc com són ”EncodingUtil” i el “TimeDiff” i també els noms de les classes "RiscossDB" i "RiscossDBDomain". La raó per la qual s'han modificat els noms i els noms d'algunes de les seves funcions és perquè els noms que tenien anteriorment causaven confusió sobre quina era la part a la qual s'estava fent referència. La classe "RiscossDBDomain" és la que gestiona als usuaris i als dominis i és la classe que s'ha implementat a la nova BD.

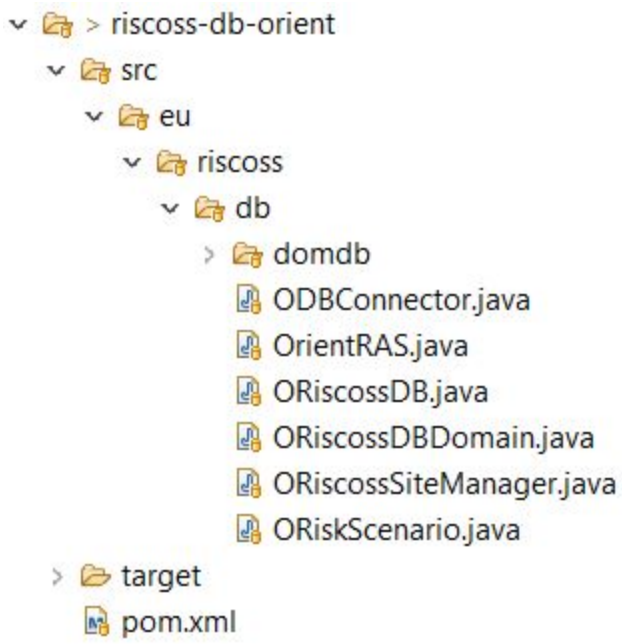


Figura 12: Package riscoss-db-orient

Per tal que els noms de les classes siguin homogènies també s'ha modificat el nom de la classe a "ORiscossDB" i "ORiscossDBDomain".

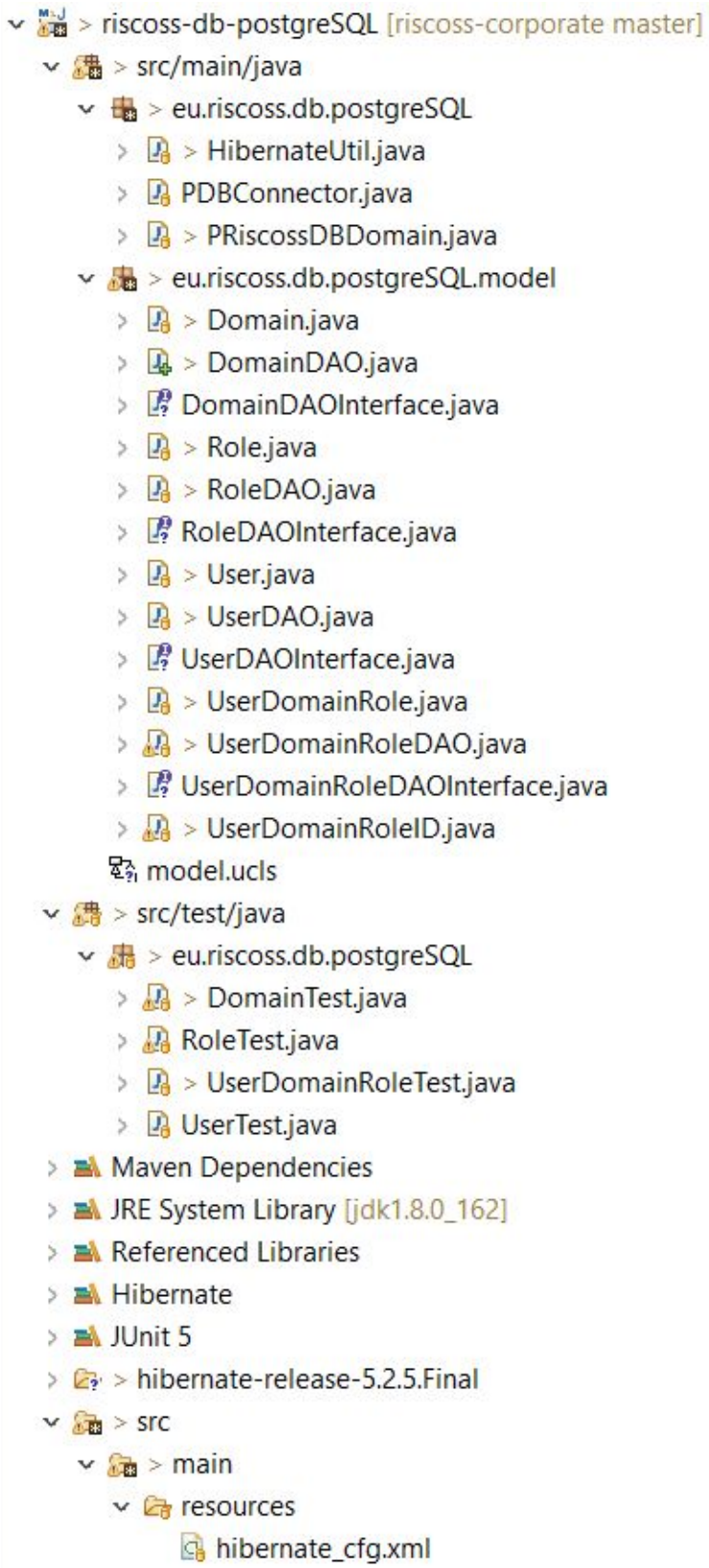


Figura 13: Package riscoss-db-postgreSQL

El package creat amb el codi de postgreSQL rep el nom de “riscoss-db-postgreSQL”. Per a la implementació de la base de dades es va escollir utilitzar l’eina Hibernate que necessita una configuració especial perquè funcioni correctament.

Com hem vist al punt “*6.4 Disseny nova BD*”, les taules que s'han decidit implementar són Domain, User, Role i UserDomainRole i per això les diferents classes tenen aquests noms.

A la carpeta src trobem els packages “eu.riscoss.db.postgreSQL” i “eu.riscoss.db.postgreSQL.model”. En el primer package, trobem la classe “HibernateUtil” que s’utilitza per a implementar la referencia a SessionFactory, que és el canal directe de comunicació entre la base de dades i la nostra aplicació. La classe PDBConnector és on hi ha les funcions per a obrir i tancar la base de dades i a més a més ens trobem amb PRiscossDBDomain on s’implementen les funcions de la interfície DBDomain.

Tal com podem veure a la imatge, el següent package és on trobem les entities que defineixen les taules de la base de dades i les interfícies i els DAOs amb les operacions CRUD (Create, Read, Update, Delete) necessàries.

A la carpeta src també tenim tests per provar que les funcions implementades funcionen de manera correcta.

Finalment, a la carpeta resources podem veure un fitxer “hibernate_cfg” que conté els paràmetres de configuració per a la connexió i creació de la BD.

7.2.3 Hibernate

En la configuració d'Hibernate es necessita un fitxer de configuració amb les dades de connexió i la ruta de mapeig de classes. Com podem veure, les propietats username, password i url són les dades que necessita per connectar-se a la BD. La propietat "hbm2ddl.auto" que conté un create indica que en cas que no existeixin les taules, les crea. Finalment, s'indiquen les taules on estan les classes amb la definició del contingut de les taules.

```
<?xml version="1.0" encoding="utf-8"?>
<hibernate-configuration>
  <session-factory>
    <!-- Database connection settings -->
    <property name="hibernate.dialect">org.hibernate.dialect.PostgreSQL82Dialect</property>
    <property name="hibernate.connection.driver_class">org.postgresql.Driver</property>
    <property name="hibernate.connection.username">postgres</property>
    <property name="hibernate.connection.password">admin</property>
    <property
name="hibernate.connection.url">jdbc:postgresql://127.0.0.1:5432/riscoss-db-postgres</property>
    <!-- Drop and re-create the database schema on startup -->
    <property name="hbm2ddl.auto">create</property>
    <property name="hibernate.show_sql">false</property>
        <property name="hibernate.format_sql">true</property>
        <property name="use_sql_comments">true</property>
    <!-- MAPPINGS -->
    <mapping class="eu.riscoss.db.postgreSQL.model.Domain"/>
            <mapping class="eu.riscoss.db.postgreSQL.model.User"/>
            <mapping class="eu.riscoss.db.postgreSQL.model.UserDomainRole"/>
            <mapping class="eu.riscoss.db.postgreSQL.model.Role"/>
  </session-factory>
</hibernate-configuration>
```

```
package eu.riscoss.db.postgreSQL;
public class HibernateUtil {
   private static final SessionFactory sessionFactory = buildSessionFactory();

   private static synchronized SessionFactory buildSessionFactory() {
      // read configuration and build session factory
      final StandardServiceRegistry registry =
            new StandardServiceRegistryBuilder()
                  .configure("hibernate_cfg.xml")
                  .build();

      try {
         return new MetadataSources(registry)
               .buildMetadata()
               .buildSessionFactory();
      }
      catch (Exception e) {
         StandardServiceRegistryBuilder.destroy(registry);
         System.exit(1);
      }
      return null;
   }

   public static SessionFactory getSessionFactory() {
      return sessionFactory;
   }

   public static Session openCurrentSessionWithTransaction()
   {
         Session currentSession = getSessionFactory().openSession();
                  currentSession.beginTransaction();
                  return currentSession;
   }

   public static void closeCurrentSessionWithTransaction(Session session) {
         session.getTransaction().commit();
         }

         public static void closeSessionFactory() {
                  // Close caches and connection pools
                  getSessionFactory().close();
         }
}
```

Per tal de definir el model de dades, necessitem classes que defineixin com són les taules de la BD, és a dir, les entities. En aquesta classe, indiquem a quina taula pertany, escrivint el nom de la taula al tag @Table. Després, s'indiquen les columnes que tindrà la taula i de quin tipus són. A més a més, la manera de marcar la clau primària és amb el paràmetre "@id". A més a més, s'han d'indicar les relacions amb les altres taules a través de tags com per exemple @ManyToOne que el que indica és que aquesta taula tindrà com a foreign key un Role, concretament un roleName. Una característica que trobem sovint en el desenvolupament de bases de dades és que ens podem trobar amb claus primàries compostes i això ho indiquem en Hibernate amb el tag @EmbeddedId. A part d'això, es creen els setters i getters.

```
@Entity
@Table(name = "userDomainRole")
public class UserDomainRole implements Serializable{
        @EmbeddedId
   private UserDomainRoleID id;
        @ManyToOne()
        private Role role;
        public Role getRole() {
           return role;
        }
        public void setRole( Role role) {
           this.role = role;
        }
   public UserDomainRole() { }
   public UserDomainRole(UserDomainRoleID id, Role role)
   {
        this.id = id;
        this.role = role;
   }
   public UserDomainRoleID getId() {
      return id;
   }
   public void setId(UserDomainRoleID id) {
      this.id = id;
   }
   @Override
   public String toString()
   {
      return "UserRole [username=" + id.getUsername() + ", domainName=" + id.getDomainName() + ",
role=" + role + "]";
   }
}
```

Com hem vist a l'anterior classe, userDomainRole té una clau primària i a part d'indicar amb un tag que és clau primària, s'ha de crear una nova classe per a definir els paràmetres de la clau primària. En el cas d'aquesta entity, la clau primària està composta de dos paràmetres els quals són foreing keys de la classe User i Domain. Com que la relació entre les classes és de molts-un, s'utilitza el tag @ManyToOne un altre cop.

```
@SuppressWarnings("serial")
@Embeddable
public class UserDomainRoleID implements Serializable {

   @ManyToOne(cascade = CascadeType.ALL)
   private User user;

   @ManyToOne(cascade = CascadeType.ALL)
   private Domain domain;

   public UserDomainRoleID() {
   }

   public UserDomainRoleID(User user, Domain domain) {
      this.user = user;
      this.domain = domain;
   }

   public User getUsername() {
      return user;
   }

   public void setUsername(User user)
   {
         this.user = user;
   }

   public Domain getDomainName() {
      return domain;
   }

   public void setDomainName(Domain domain)
   {
         this.domain = domain;
   }
}
```

Després d'analitzar diverses opcions, es va decidir que el millor era utilitzar sentencies HQL de manera que si en un futur es decideix canviar la base de dades, no s'hauran de fer canvis sintàctics a les sentències. És una manera d'abstracció per a facilitar canvis futurs. A més a més, en algunes funcions s'ha utilitzat la API Hibernate Criteria perquè facilita les consultes a través de les seves funcions. Tot i que totes les consultes es podrien haver fet amb criteria, s'ha utilitzat en aquelles funcions en les quals es necessitava fer ús del "Distinct", és a dir, que la funció no retorni valors repetits. A continuació es mostren algunes de les funcions implementades:

Descripció query	Codi
Llistar roles dels usuaris d'un domini	*public List<String> listRoles (String domainName)* *{* *Session s = HibernateUtil.getSessionFactory().openSession();* *List<String> lDomainroles = new ArrayList<String>();* *CriteriaBuilder builder = s.getCriteriaBuilder();* *CriteriaQuery<String> query = builder.createQuery(String.class);* *Root<UserDomainRole> root = query.from(UserDomainRole.class);* *query.select(root.<String>get("role")).where(builder.equal(root.get("id").get("domain").get("domainName"), domainName));* *query.distinct(true);* *Query<String> q= s.createQuery(query);* *List<String> userRoleList=q.getResultList();* *if(userRoleList != null && userRoleList.size() > 0)* *{* *for (String userRole : userRoleList)* *{* *lDomainroles.add(userRole);* *}* *}* *s.close();* *return lDomainroles;* *}*
Obtenir el role d'un usuari en un domini	*public String getRole(String username, String domainName)* *{* *Session s = HibernateUtil.getSessionFactory().openSession();* *String role = null;* *User user = (User) s.get(User.class, username);* *Domain domain = (Domain) s.get(Domain.class, domainName);* *TypedQuery<Role> query = s.createQuery("Select role from UserRole ur where ur.id.user= :user and ur.id.domain= :domain", Role.class);* *query.setParameter("user", user);* *query.setParameter("domain", domain);* *List<Role> userRole = query.getResultList();* *if(userRole != null && userRole.size() > 0)*

	{ *role = userRole.get(0).getRoleName();* *}* *s.close();* *return role;* *}*
Guardar un UserDomainRole	*public void save(UserDomainRole entity)* *{* *Session s = HibernateUtil.getSessionFactory().openSession();* *Transaction tx = s.beginTransaction();* *s.save(entity);* *s.flush();* *tx.commit();* *s.close();* *}*

Taula 10: Exemples operacions CRUD

7.2.4 PostgreSQL

En aquest apartat es mostren les taules que es generen a partir de la configuració d'Hibernate.

Nom taula	Create table
domain	*CREATE TABLE public.domain* *(* *domainname character varying(255) COLLATE pg_catalog."default" NOT NULL,* *isprivate boolean,* *role_rolename character varying(255) COLLATE pg_catalog."default",* *CONSTRAINT domain_pkey PRIMARY KEY (domainname),* *CONSTRAINT role_rolename_fkey FOREIGN KEY (role_rolename)* *REFERENCES public.role (rolename)* *)*
user	*CREATE TABLE public."user"* *(* *username character varying(255) COLLATE pg_catalog."default" NOT NULL,* *firstname character varying(255) COLLATE pg_catalog."default",* *issuperadmin boolean,* *lastname character varying(255) COLLATE pg_catalog."default",* *password character varying(255) COLLATE pg_catalog."default",* *CONSTRAINT user_pkey PRIMARY KEY (username)* *)*
userDomanRole	*CREATE TABLE public.userrole* *(* *domain_domainname character varying(255) COLLATE pg_catalog."default" NOT NULL,* *user_username character varying(255) COLLATE pg_catalog."default" NOT NULL,* *role_rolename character varying(255) COLLATE pg_catalog."default",* *CONSTRAINT userrole_pkey PRIMARY KEY (domain_domainname, user_username),* *CONSTRAINT role_rolename_fkey FOREIGN KEY (role_rolename)* *REFERENCES public.role (rolename)* *CONSTRAINT user_username_fkey FOREIGN KEY (user_username)* *REFERENCES public."user" (username)* *CONSTRAINT domain_domainname_fkey FOREIGN KEY (domain_domainname)* *REFERENCES public.domain (domainname)* *}*
role	*CREATE TABLE public.role* *(* *rolename character varying(255) COLLATE pg_catalog."default" NOT NULL,* *CONSTRAINT role_pkey PRIMARY KEY (rolename)* *)*

Taula 11: Codi creació taules bases de dades PostgreSQL

8. Proves

Una part important del desenvolupament és la realització de proves per tal d'assegurar el correcte funcionament del sistema. D'aquesta manera, es comprova el compliment dels requisits funcionals i no funcionals. La plataforma RISCOSS tenia definits casos de prova per provar les diverses funcionalitats del sistema, i s'han afegit de nous per poder testejar la nova implementació.

A l'apartat "14. Annex" es poden trobar els casos de prova que hi havia abans de fer aquest projecte i a continuació es mostren les noves proves fetes que s'han definit per comprovar la connexió a la base de dades i les sentencies HQL que fan les operacions CRUD corresponents a les taules "user", "domain", “userdomainrole” i "role" que hi ha a la base de dades PostgreSQL.

Tag Test	Descripció	Codi
PTest-01	Crear domain	*Domain domain1 = new Domain("Maria's Domain", role1, true);* *Domain domain2 = new Domain("Elena's Domain", role1, false);* *Domain domain3 = new Domain("Joan's Domain", role2, false);* *Domain domain4 = new Domain("Marc's Domain", role3, true);* *domainDAO.save(domain1);* *domainDAO.save(domain2);* *domainDAO.save(domain3);* *domainDAO.save(domain4);*
PTest-02	Eliminar domain	*domainDAO.delete(domain1.getDomainName());*
PTest-03	Crear user	*User user1 = new User("username_maria", "mariaPassw", "Maria","Vives", false);* *User user2 = new User("username_laura", "lauraPassw", "Laura","Vives", false);* *User user3 = new User("username_carlos", "carlosPassw", "Carlos","Vives", false);* *User user4 = new User("username_roberto", "robertoPassw", "Roberto","Vives", false);* *User user5 = new User("admin", "admin", "admin","admin", false);* *userDao.save(user1);* *userDao.save(user2);* *userDao.save(user3);* *userDao.save(user4);* *userDao.save(user5);*
PTest-04	Eliminar user	*userDao.delete(user4.getUserName());* *userDao.delete(user2.getUserName());*
PTest-05	Llistar domains	*lAllDomains = domainDAO.findAll();* *System.out.println("All domains are :");*

		if(lAllDomains != null && lAllDomains.size() > 0) *for (String domains : lAllDomains) {* *System.out.println("-" + domains);* *}*
PTest-06	Llistar domains publics	*lAllPublicDomains = domainDAO.findAllPublic();* *if(lAllPublicDomains != null && lAllPublicDomains.size() > 0)* *for (String publicDomains : lAllPublicDomains) {* *System.out.println("-" + publicDomains);* *}*
PTest-07	Existeix domain	*Boolean existsDomain = domainDAO.existsDomain(domain3.getDomainName());*
PTest-08	Llegir role predefinit d'un domain	*String predefinedRole = domainDAO.getPredefinedRole(domain4.getDomainName());*
PTest-09	Crear role	*roleDao.createRole("admin");* *roleDao.createRole("consumer");* *roleDao.createRole("modeler");* *roleDao.createRole("producer");*
PTest-10	Llistar rols	*lRoles = roleDao.listRoles();* *if(lRoles != null && lRoles.size() > 0)* *for (String roles : lRoles) {* *System.out.println("-" + roles);* *}*
PTest-11	Eliminar role	*roleDao.delete("consumer");*
PTest-12	User és admin	*Boolean user1isAdmin = userDao.isAdmin(user1.getUserName());*
PTest-13	Llistar users donat un pattern	*lUsers = userDao.listUsers("username");* *if(lUsers != null && lUsers.size() > 0)* *for (String user : lUsers) {* *System.out.println("-" + user);* *}*
PTest-14	Comprovar si dues contrasenyes són iguals	*Boolean isSamePass = userDao.checkPassword(user1.getUserName(), user1.getPassword());*
PTest-15	Llistar tots els users	*lUsers = userDao.listUsers();* *if(lUsers != null && lUsers.size() > 0)* *for (String user : lUsers) {* *System.out.println("-" + user);* *}*
PTest-16	Crear userDomainRole	*UserDomainRole userRole1 = new UserDomainRole(new UserDomainRoleID(user1, domain1), role4);*

		userRoleDao.save(userRole1);
PTest-17	Llistar els users que tenen el role assignat per paràmetre	*lUsers = userRoleDao.listUsers("producer")* *if(lUsers != null && lUsers.size() > 0)* *for (String user : lUsers) {* *System.out.println("-" + user);* *}*
PTest-18	Llistar domains que tenen assignat l'username passat per paràmetre	*lDomains = userRoleDao.listDomains(user1.getUserName());* *if(lDomains != null && lDomains.size() > 0)* *for (String domain : lDomains) {* *System.out.println("-" + domain);* *}*
PTest-19	Llistar els roles que tenen els users d'un domain	*lRoles = userRoleDao.listRoles(domain4.getDomainName());* *if(lRoles != null && lRoles.size() > 0)* *for (String role : lRoles) {* *System.out.println("-" + role);* *}*
PTest-20	Eliminar userDomainRole	*userRoleDao.delete(userRole1);*
PTest-21	Elimina un user d'un domain	*userRoleDao.removeUserFromDomain (user1.getUserName(), domain1.getDomainName());*
PTest-22	Assigna a un user i un domain un role	*userRoleDao.setUserRole ("username_maria", "admin", "Maria's Domain")*

Taula 12: Tests de proves queries PostgreSQL

La metodologia seguida per a la realització de tests ha sigut prova i error. El que es va fer primer és la creació dels tests per tenir una primera visió de quines funcions s'havien d'implementar i més tard es va fer la implementació. Un cop es va finalitzar la implementació, es van passar els tests per comprovar el correcte funcionament de les operacions implementades. A continuació es mostren els resultats i les observacions per cada comprovació feta.

Test	Data	Resultat	Observacions
PTest-01	01-04-2018 02-04-2018 03-04-2018	No passat No passat Passat	Problemes amb la configuració del fitxer hibernate_cfg
PTest-02	03-04-2018	Passat	
PTest-03	03-04-2018	Passat	
PTest-04	03-04-2018	Passat	

PTest-05	01-04-2018 02-04-2018 03-04-2018 04-04-2018 05-04-2018	No passat No passat No passat No passat Passat	No passava a causa que el List que s'utilitza dins la funció no estava inicialitzat i semblava error de configuració d'Hibernate
PTest-06	01-04-2018 02-04-2018 03-04-2018 04-04-2018 05-04-2018	No passat No passat No passat No passat Passat	Igual que al PTest-05
PTest-07	07-04-2018	Passat	
PTest-08	07-04-2018	Passat	
PTest-09	07-04-2018	Passat	
PTest-10	07-04-2018	Passat	
PTest-11	08-04-2018	Passat	
PTest-12	08-04-2018	Passat	
PTest-13	07-04-2018 08-04-2018 09-04-2018 10-04-2018	No passat No passat No passat Passat	Problemes per a crear una crida passant un pattern, és a dir, per a trobar la similitud. A l'inici no funcionava, després només agafava els users que començaven com el pattern i finalment es va trobar la solució correcta.
PTest-14	09-04-2018 10-04-2018 11-04-2018	No passat No passat Passat	Primer es guardaven les contrasenyes sense encriptar. El següent pas va ser encriptar la contrasenya i comparar-la amb la contrasenya encriptada donada. La complicació va ser trobar la manera d'encriptar.
PTest-15	08-04-2018	Passat	
PTest-16	08-04-2018	Passat	
PTest-17	08-04-2018	No passat	Aparentment semblava

	09-04-2018 10-04-2018 11-04-2018 12-04-2018	No passat No passat No passat Passat	que funcionava fins que la desenvolupadora es va adonar que els resultats sortien repetits. És a dir, es necessitava fer un distinct per a evitar repeticions en els resultats. Es va decidir utilitzat aquesta manera per a poder indicar el retorn de resultats amb distinct de manera fàcil. També hi va haver problemes amb com s'ha d'accedir a un paràmetre d'una clau composta.
PTest-18	08-04-2018 09-04-2018 10-04-2018 11-04-2018 12-04-2018	No passat No passat No passat No passat Passat	Igual que al PTest-17
PTest-19	08-04-2018 09-04-2018 10-04-2018 11-04-2018 12-04-2018	No passat No passat No passat No passat Passat	Igual que al PTest-17
PTest-20	08-04-2018	Passat	
PTest-21	12-04-2018	Passat	
PTest-22	12-04-2018	Passat	

Taula 13: Evolució proves queries PostgreSQL

9. Obstacles

Durant la realització del TFG han anat sorgint diversos obstacles relacionats amb les fases l'anàlisi i disseny i d'implementació. A continuació, es detallen les dificultats i solucions.

9.1 Configuració entorn

La configuració de l'entorn de treball ha sigut problemàtica per diverses raons. La descripció dels passos a seguir per descarregar els projectes necessaris i instal·lar els components i llibreries per compilar era incomplet. A part d'això, fer el "build" i fer el desplegament de l'aplicació també va ser complicat a causa del desconeixement de les tecnologies utilitzades i problemes amb la compatibilitat de les versions d'alguns dels components. Per tal de poder avançar en aquest aspecte, es va demanar ajuda a una persona que havia participat en l'implementació del projecte.

9.2 Familiarització amb estructura del projecte

El primer pas que es va fer va ser informar-se de què feia la plataforma i quines eren les funcionalitats que oferia a partir de la documentació que hi ha a la web de la plataforma i la wiki de GitHub. Un cop fet això, entendre com estava estructurat el codi i quina part s'encarregava de cada funcionalitat va ser difícil per la complexitat que té el projecte. A més a més, s'havia d'estudiar quines parts eren les afectades i on s'havien de fer els canvis. A causa de la complexitat mencionada, el temps utilitzat per realitzar aquesta tasca va ser major que la planificació estimada inicial.

9.3 Compatibilitat entre components

En incorporar nous components, Hibernate i PostgreSQL, amb les seves últimes versions va fer que hi hagués problemes de compatibilitat amb alguns dels components que ja hi havia. Com que no es trobava cap solució, es van provar diverses maneres d'estructurar els fitxers de configuració Hibernate i diverses versions fins que finalment va funcionar.

9.4 Hibernate i el llenguatge HQL

A la fase d'implementació es va decidir utilitzar l'eina Hibernate i el llenguatge HQL per tal de facilitar de cara al futur els desenvolupaments. Les problemàtiques que hi ha hagut estan relacionades amb l'estructura que han de tenir els fitxers i el llenguatge HQL (Hibernate query language).

L'estructura dels fitxers amb Hibernate és molt concreta i s'ha de conèixer en profunditat com es relacionen les classes amb les taules de la base de dades i quins fitxers de configuració són necessaris perquè aquesta es pugui connectar amb èxit. El desenvolupador havia tingut contacte amb Hibernate a l'assignatura d'AS (Arquitectura del Software) però haver de fer la configuració per a un entorn concret ha requerit temps d'aprenentatge.

El llenguatge HQL és concret i a part d'haver requerit un temps d'aprenentatge, s'ha necessitat més temps per problemes de compatibilitat amb la versió "Hibernate 5" i la manera en la qual s'implementen les sentències HQL.

9.5 Creació tests operacions HQL

Inicialment es volien fer proves unitàries per provar si les crides a la base de dades funcionaven correctament. Es va estar estudiant quina era la millor manera de fer-les i per la complexitat que comportava, finalment, es va crear per cada taula, proves per comprovar que les operacions actualitzaven la base de dades de la manera esperada.

9.6 Centralització implementació base de dades

Finalment, la base de dades OrientDB no estava implementada totalment amb interfícies sinó que algunes funcions base feien crides directament a la base de dades.
Inicialment, per com semblava que estava el codi estructurat, es pensava que al package riscoss-db estaven totes les interfícies amb les definicions de totes les funcions que necessita l'aplicació. Més tard, es va veure que no estava dissenyat completament per en un futur poder fer modificacions i implementacions amb altres bases de dades diferents. Un exemple és quan el servei d'autentificació fa el login i el que fa és cridar directament a una funció específica d'una classe d'OrientDB.

10. Conclusions

10.1 Resolució d’objectius

En aquest apartat veurem si els objectius inicials del projecte s'han assolit. Els objectius, tal com s'ha indicat en seccions anteriors són fer l'anàlisi de l'arquitectura i tecnologies, identificar les possibles millores i la implementació, documentació, proves i integració de la millora.

El primer objectiu d'anàlisi de l'arquitectura i tecnologies ha sigut el que ha necessitat més temps. Això és a causa que s'ha necessitat temps per familiaritzar-se amb la plataforma i amb com estava estructurat el codi. A més a més, moure's pel codi i identificar quines classes són les que s'han de modificar ha sigut complicat a causa que és un projecte gran.

L'objectiu d'identificació de possibles millores ha sigut el més curt a causa que ja s'havien donat pautes de quines eren les possibles millores a fer. Es va decidir fer la millora a la base de dades per veure si era o no viable fer els canvis i com fer-ho.

L'últim objectiu és la implementació, prova, integració i documentació de la base de dades. Es va decidir començar a implementar una de les interfícies en les quals es guarda informació sobre usuaris, dominis i els rols que aquests tenen per a acotar la feina al TFG. Actualment, estan les sentències HQL implementades i s'han fet tests per a provar que les sentències retornen la informació que toca. Tot i això, falta implementar una interfície anomenada *SiteManager* per a poder fer la integració amb la resta de codi. De manera que no s'ha assolit l'objectiu perquè no es podrà integrar amb la resta de codi per no funcionar com s'esperava. No es pot integrar encara.

La memòria d’aquest TFG és la documentació que indica l’estat actual de la plataforma.

10.2 Canvis de planificació

En el projecte hi ha hagut canvis importants en les dates d'entrega i en la finalització de les fases respecte de la planificació inicial. La fase d'anàlisi i disseny es va allargar fins a finals de novembre i això va fer que s'endarrerís la fase d'implementació i proves. Per tal d'evitar imprevistos, es va decidir fer l'entrega del TFG a l'abril. La fase d'implementació i proves es va reprendre a principis de gener i es va continuar desenvolupant fins a l'abril.

Podem observar a la taula 14, que hi ha una diferència gran entre la fase d'anàlisi i disseny entre les hores estimades a la planificació inicial i les que s'han acabat fent. Aquesta diferència és perquè no es va plasmar de manera correcta en el càlcul d'hores que en un projecte de reenginyeria l'anàlisi i disseny és la part a la qual se li dedica més temps.

Fase	Duració estimada planificació inicial (hores)	Duració (hores)
Gestió de Projectes (GEP)	75	80
Anàlisi i disseny	135	244
Implementació i proves	190	191
Documentació	50	97
TOTAL	450	612

Taula 14: Estimació en hores de les fases del projecte inicial i final

10.3 Desviacions planificació temporal

A la secció "*14.2 Diagrama de Gantt final*" es troba el diagrama final, on es mostra el nom de la tasca, quan s'inicia i quan finalitza. Com es pot observar, la fase d'anàlisi i disseny es va allargar més del que s'esperava, fent que es decidís presentar el projecte a l'abril i així tenir temps per a desenvolupar la implementació, fer la memòria final i preparar la presentació.

10.4 Desviacions en el pressupost

A causa de desviacions en la planificació temporal inicial, hi ha hagut canvis en el pressupost que han fet que aquest hagi augmentat. Per tal de veure les desviacions, en els següents apartats es mostren a les taules el cost estimat que es va calcular en l'estimació inicial i es pot trobar a l'apartat "*4. Gestió econòmica inicial*" d'aquest document i el cost final.

10.4.1 Recursos humans finals

A les següents taules es mostra la diferència del cost dels recursos humans. La taula 15 és un resum del cost i la taula 16 és un desglossament per fases i subfases en les quals ha participat cada rol.

Com podem observar, el canvi més gran és per l'analista perquè és el rol que ha treballat a la fase d'anàlisi i disseny i és la que ha augmentat més respecte a l'estimació inicial. També trobem canvis en el programador i el cap de projecte. El cap de projecte ha disminuït a causa que encara no s'ha tingut en compte les hores dedicades al document final.

Rol	Preu per hora (€/h)	Hores previstes	Cost estimat (€)	Hores realitzades	Cost final (€)
Cap de projecte	45,00	125	5.625,00	177	7.965,00
Analista	40,00	95	3.800,00	155	6.200,00
Dissenyador	35,00	40	1.400,00	40	1.400,00
Programador	25,00	190	4.750,00	240	6.000,00
TOTAL		450	15.575,00	612	21.565,00

Taula 15: Pressupost final de recursos humans

	Temps de dedicació (en hores)				
Fase	Cap de projecte	Analista	Dissenyador	Programador	Cost final(€)
Gestió de projectes	80	0	0	0	3.600,00
Abast del projecte i contextualització	15	0	0	0	675,00
Planificació temporal	11	0	0	0	495,00
Gestió econòmica i sostenibilitat	13	0	0	0	585,00
Presentació preliminar	10	0	0	0	450,00
Plecs de condicions de les especialitats	8	0	0	0	360,00
Presentació oral i document final	23	0	0	0	1.035,00
Anàlisi i disseny	0	155	40	49	8.825,00
Instal·lació entorn i projecte	0	0	0	49	1.225,00
Anàlisi arquitectura i tecnologies actuals	0	150	0	0	6.000,00
Identificació aspectes a millorar	0	5	0	0	200,00
Disseny conceptual/classes de la part de la BD	0	0	40	0	1.400,00
Implementació i proves	0	0	0	191	4.775,00
Desenvolupament codi	0	0	0	141	3.525,00
Proves	0	0	0	50	1.250,00
Integració a la resta del projecte	0	0	0	0	0,00
Documentació	97	0	0	0	4.365,00
Memòria final	87	0	0	0	3.915,00
Preparació defensa oral	10	0	0	0	450,00
TOTAL	177	155	40	240	21.565,00

Taula 16: Costos finals directes per activitat

10.4.2 Despeses generals finals

A la taula següent es mostren els canvis en el cost final de la part de les despeses generals. Com podem veure, a causa que ha augmentat el temps 3 mesos, el cost total també ha incrementat fins a arribar als 667,10 €.

Producte	Preu(€)	Període estimat	Cost estimat(€)	Període final	Cost final (€)
Transport (T-Jove 1 zona)	35,00 €/mes	7 mesos	245,00	10 mesos	350,00
Connexió a Internet	30,00 €/mes	7 mesos	210,00	10 mesos	300,00
Consum energètic	0,14 €/KWh	450 hores	12,60	612 hores	17,10
TOTAL			467,60		667,10

Taula 17: Pressupost final de despeses generals

10.4.3 Pressupost total final

A continuació trobem el resum del pressupost i podem veure que aquest ha augmentat més del que s'esperava, i amb els imprevistos i la contingència calculada a la planificació inicial el pressupost no és suficient per cobrir les despeses.

Concepte	Cost estimat (€)	Cost final (€)
Recursos humans	15.575,00	21.565,00
Recursos hardware	101,02	101,02
Recursos software	0,00	0,00
Despeses generals	467,60	667,10
Imprevistos	44,90	-
Contingència	2.421,54	-
TOTAL	18.790,06	22.333,12

Taula 18: Pressupost total inicial i final

10.5 Futur

Des de l'inici del projecte, les decisions que s'han pres i s'han dut a terme han estat sempre focalitzades en com facilitar els futurs desenvolupaments i reduir el temps d'aprenentatge, facilitant així la participació en el projecte.

Millores que s'han fet relacionades amb el que ja hi havia són modificacions en el codi per tal de tenir totes les classes relacionades amb la base de dades en els mateixos packages, s'han canviat noms a classes perquè quedi més clar que és el que fan i s'ha mantingut la mateixa estructura de noms per al nou package de la nova base de dades.

Amb el nou codi desenvolupat s'ha fet ús de l'eina Hibernate perquè si en un futur se segueix desenvolupant la base de dades, aquesta eina facilitarà el disseny i la implementació de les funcions del sistema a causa de la complexitat que tenen.

10.6 Valoracions personals

Durant tota la carrera a les diferents assignatures els projectes que es desenvolupen es fan des de zero per l'alumne, on es posen en pràctica els coneixements adquirits d'una fase concreta del desenvolupament de software. Un dels motius pel qual vaig escollir aquest projecte és perquè era un projecte que ja estava desenvolupat i en funcionament, on s'havien de fer canvis. A part de ser una novetat que no havia fet durant la carrera, era un projecte on podria aprendre dels problemes que hi ha en projectes reals.

Haver participat en el desenvolupament d'aquesta plataforma m'ha fet adonar que realment hi ha un procés d'aprenentatge tant en l'àmbit de codi com de tecnologies i pot ser molt gran depenent de l'estat de la documentació i l'arquitectura. A més a més, que hagi passat un temps relativament curt, 2 anys, sense que hagi estat actualitzat pot donar problemes de compatibilitat amb versions posteriors dels components.

Després d'haver participat en un projecte com aquest, ets conscient de la importància que té seguir patrons de disseny, documentació, capçaleres a les funcions, etc. perquè el projecte tingui una estructura homogènia i en cas que alguna altra persona o un mateix hagi de fer modificacions en un futur aquestes siguin més fàcils de fer.

11. Competències tècniques associades

En el següent apartat es descriuran les competències tècniques associades i la seva justificació.

CES1.1: Desenvolupar, mantenir i avaluar sistemes i serveis software complexos i/o crítics. [Bastant].

> En aquest projecte s'ha hagut d'analitzar quin era l'estat actual d'una plataforma complexa per la gran quantitat de mòduls que la componen i l'escassa documentació que hi ha al respecte. A més a més, s'ha hagut d'avaluar si la millora que es volia fer era viable i el desenvolupament de nou codi, concretament una de les parts de la base de dades.

CES1.2: Donar solució a problemes d'integració en funció de les estratègies, dels estàndards i de les tecnologies disponibles. [Bastant].

> En el transcurs del projecte, les decisions que s'han pres han sigut pensant a facilitar el futur desenvolupament de les funcionalitats de la base de dades relacional. S'ha decidit utilitzar Hibernate perquè sigui més fàcil el desenvolupament i els canvis en l'àmbit de codi i de tecnologies perquè aquesta eina dóna la possibilitat de canviar de base de dades sense haver d'adaptar el codi. A més a més, s'ha creat un nou mòdul separat a la resta per tal de tenir tot independent i que sigui més fàcil de mantenir.

CES1.3: Identificar, avaluar i gestionar els riscos potencials associats a la construcció de software que es poguessin presentar. [Una mica].

> Durant la realització del projecte s'han tingut present els possibles riscos potencials que ens podíem trobar a l'hora de desenvolupar el codi. Un cop es va veure com estava estructurat el codi es va decidir realitzar una part de les funcionalitats del sistema. Més endavant i en veure que no tot estava separat com tocava, es va decidir implementar algunes parts del codi per a mostrar quin comportament hauria de tenir sent funcional completament, tenint en compte que en un futur s'hauran de fer les modificacions oportunes. Tanmateix, la configuració de l'entorn bàsica per a poder desenvolupar ha donat problemes i s'ha solucionat consultant a antics desenvolupadors de la plataforma.

CES1.5: Especificar, dissenyar, implementar i avaluar bases de dades. [En profunditat].

> En aquest TFG el que s'ha fet és dissenyar la nova base de dades i s'ha implementat i avaluat la part de gestió d'usuaris, que és la part que s'ha decidit que es duria a terme en aquest projecte.

CES1.7: Controlar la qualitat i dissenyar proves en la producció de software. [Bastant].

A causa de la complexitat del projecte, s'han fet proves per avaluar que les dades es guarden com s'espera a través de tests. Per tal d'avaluar el correcte funcionament del sistema, es farà ús d'alguns casos de test que hi havia a l'aplicació més alguns creats específicament.

CES3.1: Desenvolupar serveis i aplicacions multimèdia. [Bastant].

Tal com s'ha esmentat anteriorment, es desenvoluparan nous mòduls en un projecte software després d'haver identificat i analitzat els possibles problemes.

12. Referencies

[1] "OrientDB." https://orientdb.com/. Accedit 20 Set. 2017.
[2] "PostgreSQL." https://www.postgresql.org. Accedit 23 Set. 2017.
[3] "SQLite." https://www.sqlite.org/. Accedit 23 Set. 2017.
[4] Rentería, Miguel Ángel Sámano, and Ramón Rivera Espinosa. "Implementación del modelo integral colaborativo (MDSIC) como fuente de innovación para el desarrollo ágil de software en las empresas de la zona centro-occidente en México." (2014).http://www.eumed.net/tesis-doctorales/2014/jlcv/software.htm. Accedit 24 Set. 2017.
[5] "GitHub." https://github.com/. Accedit 30 Set. 2017.
[6] "Projectlibre." https://www.projectlibre.com/. Accedit 30 Set. 2017.
[7] "Cacoo." https://cacoo.com/. Accedit 30 Set. 2017.
[8] "ObjectAid." http://www.objectaid.com/. Accedit 30 Nov. 2017.
[9] "Google Drive." https://www.google.com/drive/. Accedit 30 Set. 2017.
[10] "Estudio de remuneración 2017 - Michael Page." https://www.michaelpage.es/sites/michaelpage.es/files/MP_SPA_ON_ER_IT_03052017.pdf. Accedit 09 Oct. 2017.
[11] "riscoss.eu: Managing Risk and Cost in Open Source Software Projects" http://www.riscoss.eu/. Accedit 28 Jul. 2017.
[12] "ISO 25010 - ISO/IEC 25000." http://iso25000.com/index.php/en/iso-25000-standards/iso-25010. Accedit 20 Mar. 2018.
[13] "Maven – Welcome to Apache Maven." https://maven.apache.org/. Accedit 04 Mar. 2018.
[14] "[GWT]." http://www.gwtproject.org/. Accedit 04 Mar. 2018.
[15] "Apache License, Version 2.0 - The Apache Software Foundation!." https://www.apache.org/licenses/LICENSE-2.0. Accedit 04 Mar. 2018..
[16] "Hibernate. Everything data. - Hibernate." http://hibernate.org/. Accedit 04 Mar. 2018.
[17] "PostgreSQL." https://www.postgresql.org/. Accedit 04 Mar. 2018.

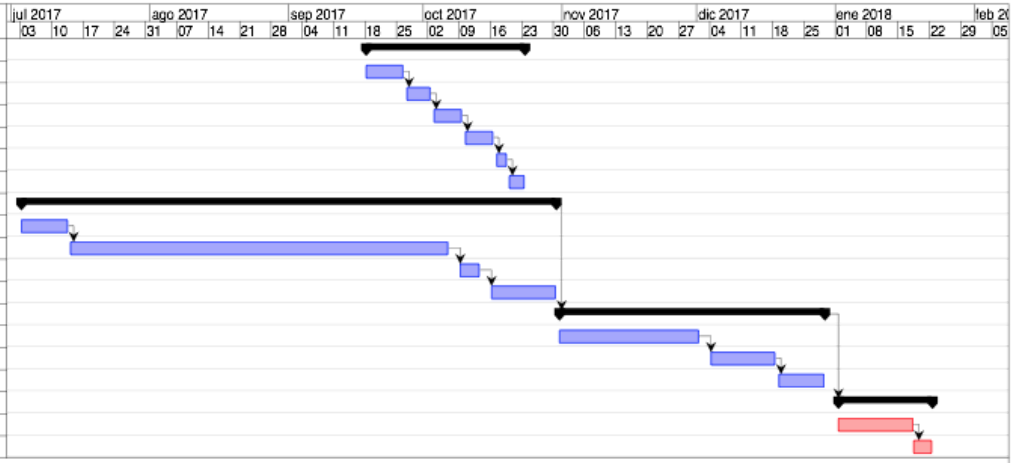
13. Bibliografia

[1] Repositori RISCOSS a GitHub [En línia] [Consulta: 4 set. 2017]
<https://github.com/RISCOSS>
[2] Pàgina web RISCOSS [En línia] [Consulta: 4 set. 2017]
<http://www.riscoss.eu>
[3] Informació de les fases del projecte RISCOSS [En línia] [Consulta: 20 jul. 2017]
<http://www.riscoss.eu/bin/view/Share/Public_Deliverables>
[4] Pàgina FIB [En línia] [Consulta: 1 oct. 2017]
<https://www.fib.upc.edu/ >

14. Annex

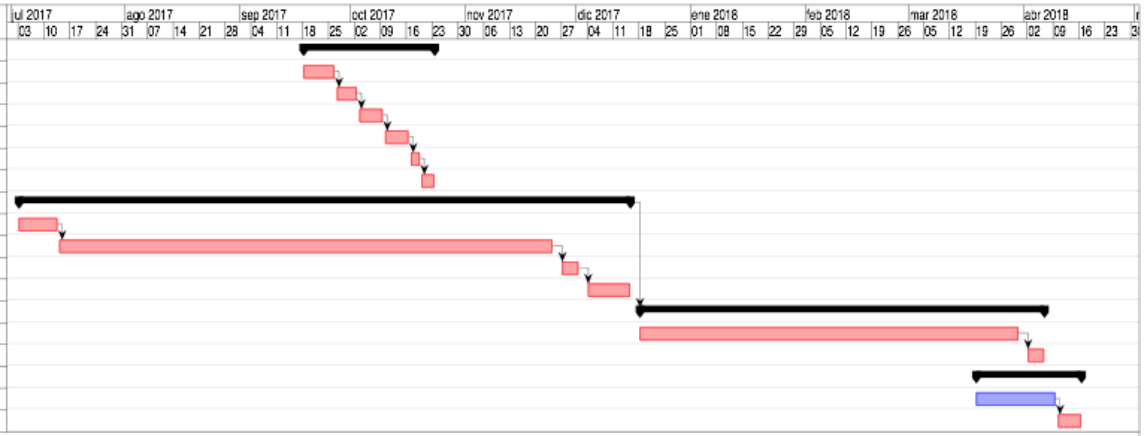
14.1 Diagrama de Gantt inicial

	Nombre	Inicio	Terminado
1	**Gestió de Projectes (GEP)**	**18/09/17 8:00**	**23/10/17 17:00**
2	Abast del projecte i contextualització	18/09/17 8:00	26/09/17 17:00
3	Planificació temporal	27/09/17 8:00	2/10/17 17:00
4	Gestió econòmica i sostenibilitat	3/10/17 8:00	9/10/17 17:00
5	Presentació preliminar	10/10/17 8:00	16/10/17 17:00
6	Plecs de condicions de les especialitats	17/10/17 8:00	19/10/17 17:00
7	Presentació oral i document final	20/10/17 8:00	23/10/17 17:00
8	**Anàlisi i disseny**	**3/07/17 8:00**	**30/10/17 17:00**
9	Instal·lació entorn i projecte	3/07/17 8:00	13/07/17 17:00
10	Anàlisi arquitectura i tecnologies actuals	14/07/17 8:00	6/10/17 17:00
11	Identificació aspectes a millorar	9/10/17 8:00	13/10/17 17:00
12	Disseny conceptual/classes de la part de la BD	16/10/17 8:00	30/10/17 17:00
13	**Implementació i proves**	**31/10/17 8:00**	**29/12/17 17:00**
14	Desenvolupament codi	31/10/17 8:00	1/12/17 17:00
15	Proves	4/12/17 8:00	18/12/17 16:00
16	Integració a la resta del projecte	19/12/17 9:00	29/12/17 17:00
17	**Documentació**	**1/01/18 15:00**	**22/01/18 17:00**
18	Memòria final	1/01/18 15:00	18/01/18 14:00
19	Preparació defensa oral	18/01/18 14:00	22/01/18 17:00



14.2 Diagrama de Gantt final

	Nombre	Inicio	Terminado
1	**Gestió de Projectes (GEP)**	**18/09/17 8:00**	**23/10/17 17:00**
2	Abast del projecte i contextualització	18/09/17 8:00	26/09/17 17:00
3	Planificació temporal	27/09/17 8:00	2/10/17 17:00
4	Gestió econòmica i sostenibilitat	3/10/17 8:00	9/10/17 17:00
5	Presentació preliminar	10/10/17 8:00	16/10/17 17:00
6	Plecs de condicions de les especialitats	17/10/17 8:00	19/10/17 17:00
7	Presentació oral i document final	20/10/17 8:00	23/10/17 17:00
8	**Anàlisi i disseny**	**3/07/17 8:00**	**15/12/17 17:00**
9	Instal·lació entorn i projecte	3/07/17 8:00	13/07/17 17:00
10	Anàlisi arquitectura i tecnologies actuals	14/07/17 8:00	24/11/17 17:00
11	Identificació aspectes a millorar	27/11/17 8:00	1/12/17 17:00
12	Disseny conceptual/classes de la part de la BD	4/12/17 8:00	15/12/17 17:00
13	**Implementació i proves**	**18/12/17 8:00**	**6/04/18 17:00**
14	Desenvolupament codi	18/12/17 8:00	30/03/18 17:00
15	Proves	2/04/18 8:00	6/04/18 17:00
16	**Documentació**	**19/03/18 9:00**	**16/04/18 17:00**
17	Memòria final	19/03/18 9:00	9/04/18 17:00
18	Preparació defensa oral	10/04/18 13:00	16/04/18 17:00



14.3 Casos de prova - TEI1

TC Tag	Descripció
SC-TEI-01	Creació d'una comunitat OSS a la plataforma RISCOSS
SC-TEI-02	Creació d'un component OSS a la plataforma RISCOSS
SC-TEI-03	Creació d'un producte a la plataforma RISCOSS
SC-TEI-04	Creació d'una "release" a la plataforma RISCOSS
SC-TEI-05	Vista de la comunitat OSS creada a la plataforma RISCOSS
SC-TEI-06	Búsqueda de la comunitat OSS a la plataforma RISCOSS
SC-TEI-07	Vista d'un component OSS a la llista de components OSS creats a la plataforma RISCOSS
SC-TEI-08	Búsqueda del component OSS a la plataforma RISCOSS
SC-TEI-09	Vista del producte creat a la plataforma RISCOSS
SC-TEI-10	Búsqueda del producte a la plataforma RISCOSS
SC-TEI-11	Vista de la "release" creada a la plataforma RISCOSS
SC-TEI-12	Búsqueda de la "release" a la plataforma RISCOSS
SC-TEI-13	Modificar una comunitat OSS
SC-TEI-14	Modificar un component OSS
SC-TEI-15	Modificar un producte
SC-TEI-16	Modificar una "release"
SC-TEI-17	Eliminar una comunitat OSS
SC-TEI-18	Eliminar un component OSS
SC-TEI-19	Eliminar un producte
SC-TEI-20	Eliminar una "release"
SC-TEI-21	Creació d'un risk model a la plataforma RISCOSS
SC-TEI-22	Vista del risk model creat a la plataforma RISCOSS

SC-TEI-23	Búsqueda d'un risk model
SC-TEI-24	Modificar un risk model existent
SC-TEI-25	Eliminar un risk model existent
SC-TEI-26	Creació d'un indicador de risc a la platagorma RISCOSS
SC-TEI-27	Vista de l'indicador de risc creat a la platagorma RISCOSS
SC-TEI-28	Búsqueda d'un indicador de risc
SC-TEI-29	Modificar un indicador de risc
SC-TEI-30	Eliminar un indicador de risc
SC-TEI-31	Creació d'una linea de producte SW amb dues capes (components OSS i comunitats OSS) a la plataforma RISCOSS
SC-TEI-32	Creació d'una linea de producte SW amb tres capes (components OSS, comunitats OSS i productes) a la plataforma RISCOSS
SC-TEI-33	Creació d'una linea de producte SW amb quatre capes (components OSS, comunitats OSS, productes i "release") a la plataforma RISCOSS
SC-TEI-34	Vista d'una linea de producte SW amb quatre capes (components OSS, comunitats OSS, productes i "release") a la plataforma RISCOSS
SC-TEI-35	Búsqueda d'una linea de producte SW amb quatre capes (components OSS, comunitats OSS, productes i "release") a la plataforma RISCOSS
SC-TEI-36	Modificació d'una linea de producte SW amb quatre capes (components OSS, comunitats OSS, productes i "release") a la plataforma RISCOSS
SC-TEI-37	Eliminació d'una linea de producte SW amb quatre capes (components OSS, comunitats OSS, productes i "release") a la plataforma RISCOSS
SC-TEI-38	Associar un risk model a un objecte dins d'una linea de producte SW amb 4 capes (components OSS, comunitats OSS, productes, "release")
SC-TEI-39	Eliminar l'associació d'un risk model a un objecte dins d'una linea de producte SW amb 4 capes (components OSS, comunitats OSS, productes, "release")
SC-TEI-40	Verificació de la sensibilitat de "license risk evaluation" per a afegir variacions

SC-TEI-41	Verification of the sensibility of quality risk evaluation to input data variation
SC-TEI-42	Testejar la fiabilitat de la llicencia d'evaluació de risc a través d'una comparació visual entre les dades donades i l'evaluació de RISCOSS
SC-TEI-43	Testejar la fiabilitat de la qualitat d'evaluació de risc a través d'una comparació visual entre les dades donades i l'evaluació de RISCOSS
SC-TEI-44	Verificació dels pasos d'instal.lació
SC-TEI-45	Configuració RISCOSS
SC-TEI-46	Usabilitat RISCOSS
SC-TEI-47	Verificació de la sensibilitat de la qualitat d'evaluació de risc Multi Layer
SC-TEI-48	Verificació de la sensibilitat dels Models (What-IF analysis)
SC-TEI-49	Mostreig de les dades
SC-TEI-50	Visibilitat de la RISCOSS session a cada usuari
SC-TEI-51	Differents user roles en una RISCOSS session
SC-TEI-52	Pujar documents en un Risk Model
SC-TEI-53	Herencia de la informació contextual de Layer a Entity
SC-TEI-54	Gestió d'un número gran d'Entities
SC-TEI-55	Pujar documents per a una entitat
SC-TEI-56	Testejar la fiabilitat de manteniment d'evaluació de risc a través d'una comparació visual entre les dades donades i l'evaluació de RISCOSS utilitzant un maintenance risk model
SC-TEI-57	Testejar la fiabilitat de seguretat d'evaluació de risc a través d'una comparació visual entre les dades donades i l'evaluació de RISCOSS utilitzant un risk model de seguretat
SC-TEI-58	Testejar la fiabilitat de la qualitat d'evaluació de risc a través d'una comparació visual entre les dades donades i l'evaluació de RISCOSS utilitzant un quality risk model
SC-TEI-59	Testejar la fiabilitat de manteniment d'evaluació de risc a través d'una comparació visual entre les dades donades i l'evaluació de RISCOSS utilitzant un maintenance risk model

SC-TEI-60	Fer l'autentificació d'un usuari a través de la API
SC-TEI-61	Testejar la creació d'una entity a través de la API
SC-TEI-62	Configuració d'un data collector a través de la API (GitHub)
SC-TEI-63	Executar data collector a través de la aPI (GitHub)
SC-TEI-64	Crear analisis a través de la API
SC-TEI-65	Executar analisis a través de la API
SC-TEI-66	Portar resultats de l'analisi a través de la API
SC-TEI-68	Eliminar una linea de producte SW creada anteriorment amb 2 capes (components OSS, comunitats OSS)
SC-TEI-69	Eliminar un Object amb relació pare/fill
SC-TEI-70	Modificar la capa pare en una linea de producte SQ amb dues o més capes (components OSS, comunitats OSS)
SC-TEI-71	Modificar el nom d'una comunitat OSS que estigui creada
SC-TEI-72	Modificar la representació jerarquica en una linea de producte SW
SC-TEI-73	Modificar la representació d'una linea de producte SW
SC-TEI-74	Búsqueda d'una Risk Analysis Session

14.4 Casos de prova - CENATIC

TC Tag	Description
SC-CEN-1	Instal.lació i configuració de la plataforma RISCOSS al data-center de CENATIC
SC-CEN-2	La plataforma hauria de permetre crear una nova entity/component
SC-CEN-3	La plataforma hauria de permetre configurar una entity/component
SC-CEN-4	La plataforma hauria de permetre eliminar entities.Una entity és un component de software registrat
SC-CEN-5	La plataforma hauria de permetre crear un nou projecte. Un projecte haura d'entendres com un conjunt de components, que són en un conjunt un producte.
SC-CEN-6	La plataforma hauria de permetre configurar un projecte existent.
SC-CEN-7	La plataforma hauria de permetre eliminar projectes.
SC-CEN-8	La plataforma hauria de permetre crear un risk model
SC-CEN-9	És necessari que la plataforma permeti modificar o configurar un risk model que estigui registrat
SC-CEN-10	La plataforma hauria de permetre eliminar risk models
SC-CEN-11	La plataforma hauria de permetre crear un nou goal model
SC-CEN-12	La plataforma hauria de permetre configurar goal models registrats
SC-CEN-13	La plataforma hauria de permetre eliminar goal models
SC-CEN-14	La plataforma hauria de permetre modificar a mà les dades donades pels data collectors dels risk models
SC-CEN-15	La plataforma hauria de permetre crear una risk session. Les risk sessions són utilitzades com a configuració d'un projecte, i els models i goal models per a fer anàlisis.
SC-CEN-16	La plataforma hauria de permetre modificar/configurar una risk session
SC-CEN-17	La plataforma hauria de permetre eliminar risk sessions

SC-CEN-18	La plataforma hauria de permetre fer run de les risk sessions
SC-CEN-19	La plataforma hauria de permetre operar en mode simulació
SC-CEN-20	Testejar una risk analysis de llicència utilitzant Activae project, conegut per CENATIC. Aquest projecte té coses peculiars com ara llicències incompatibles, components rellicenciats, etc.

14.5 Casos de prova - OW2-OSS

TC Tag	Description
SC-OW2-101	Verificació de la sensibilitat de la llicència d'evaluació de risc per a la variació de dades d'entrada.Diferents projectes haurien de retornar resultats diferents.
SC-OW2-102	Verificació de la sensibilitat de l'avaluació de qualitat de risc per a la variació de dades d'entrada.Diferents projectes haurien de retornar resultats diferents.
SC-OW2-103	Verificació de la sensibilitat de l'avaluació de activitat de risc per a la variació de dades d'entrada.Diferents projectes haurien de retornar resultats diferents.
SC-OW2-104	Desplegament de la plataforma RISCOSS a la infraestructura OW2 i la integració amb les eines OW2 SQuAT.
SC-OW2-105	Verificar que la plataforma garanteix els accessos depenent del rol d'usuari
SC-OW2-106	Assegurar que OW2 pot afegir projectes i components a la plataforma
SC-OW2-107	Assegurar que OW2 pot oferir perfils d'usuari prefets als usuaris per escollir-ne.
SC-OW2-108	Assegurar que un usuari pot fer un risk analysis en un projecte/component de OW2
SC-OW2-109	Donar resultats significatius i fàcils d'utilitzar als usuaris
SC-OW2-110	Testejar la usabilitat i la facilitat d'utilització del GitHubAnalyzer quan està loggejat
SC-OW2-111	Testejar la usabilitat i la facilitat d'utilització del GitHubAnalyzer quan està loggejat
SC-OW2-112	Testejar la usabilitat i la facilitat d'utilització del GitHubAnalyzer quan està loggejat i quan no
SC-OW2-113	Testejar la usabilitat i la facilitat d'utilització del GitHubAnalyzer
SC-OW2-14	Monitorejar les visites al GitHubAnalyzer

SC-OW2-15	Testejar la usabilitat i la facilitat d'utilització del GitHubAnalyzer
SC-OW2-16	Testejar la usabilitat i la facilitat d'utilització del GitHubAnalyzer
SC-OW2-17	Comparar els resultats de diferents models i escollir el més apropiat
SC-OW2-18	Permetre actualitzar els risk models disponibles per a que es mostri al risk analysis del OSS o crear de nous
SC-OW2-19	Fer la computació de risk transparent i fàcil d'entendre.
SC-OW2-21	Donar representació visual als risks
SC-OW2-22	Comparar els resultats de diferents models i escollir el més apropiat
SC-OW2-23	Permetre als usuaris avançats a crear els seus propis models de risk
SC-OW2-24	Permetre als usuaris actualitzar els models de risk disponibles
SC-OW2-25	Fer possible evolucionar el risk meta-model utilitzat
SC-OW2-26	Permetre als visitants entendre i revisar el complet fluxe de treball utilitzat per a calcular els indicadors de risc
SC-OW2-27	Permetre als visitants llegir un risk analysis en paper
SC-OW2-28	Validar la usabilitat de la plataforma desde telefons mobils
SC-OW2-29	Assegurar-se que la plataforma no té impacte negatiu en el medi ambient
SC-OW2-30	Assegurar que la plataforma pot escalar suficentment.
SC-OW2-31	Creació de diverses contes d'usuari
SC-OW2-32	Accés concurrent a la plataforma per diversos usuaris
SC-OW2-33	Accedir a la pàgina d'administrador i crear i eliminar projectes
SC-OW2-34	Usuaris loggejats poden tornar a fer un anàlisis en un projecte existent
SC-OW2-35	Crear, modificar, eliminar data collectors
SC-OW2-36	Fer possible la comparació de risk analysis en una escala comuna
SC-OW2-37	Adaptar la funció de normalització a noves necessitats
SC-OW2-38	Facilitat la adopció i ús de la plataforma

SC-OW2-39	Canviar el disseny dels panels dels projectes
SC-OW2-40	Accedir a les dades produïdes pels data collectors
SC-OW2-41	Verificar que les llicencies utilitzades per la plataforma són open-source
SC-OW2-42	Facilitat la difusió de la plataforma
SC-OW2-43	Accedir als informes amb les estadistiques de les pagines mes visitades, l'origen dels visitants i el temps que han estat
SC-OW2-44	Verificar que la plataforma no pot ser reconfigurada per algú que no sigui admin

14.6 Casos de prova - Moodbile-OSS

TC Tag	Description
SC-UPC-02	Verificar la sensibilitat de la license risk evaluation
SC-UPC-03	Verificar la sensibilitat de la quality risk evaluation
SC-UPC-04	Testejar l'escenari Moodbile a la plataforma
SC-UPC-05	Crear una nova entiat per a un component OSS
SC-UPC-06	Linkejar un repositori GitHub a un component OSS
SC-UPC-07	Confirmar la importació de dades de GitHub
SC-UPC-08	Carregar Risk Model